2017-01-01

Computational methods for prediction and classification of G protein-coupled receptors

Khodeza Begum
University of Texas at El Paso, kbegum@miners.utep.edu

COMPUTATIONAL METHODS FOR PREDICTION AND CLASSIFICATION OF G PROTEIN-COUPLED RECEPTORS

KHODEZA BEGUM

Master's Program in Computational Science

APPROVED:

Ming-Ying Leung, Ph.D., Chair

Rachid Skouta, Ph.D.

Xiaogang Su, Ph.D.

Charlotte M. Vines, Ph.D.

Charles Ambler, Ph.D.
Dean of the Graduate School

COMPUTATIONAL METHODS FOR PREDICTION AND CLASSIFICATION OF G PROTEIN-COUPLED RECEPTORS

by

KHODEZA BEGUM, B.S.

THESIS

Presented to the Faculty of the Graduate School of

The University of Texas at El Paso

in Partial Fulfillment

of the Requirements

for the Degree of

MASTER OF SCIENCE

COMPUTATIONAL SCIENCE PROGRAM

THE UNIVERSITY OF TEXAS AT EL PASO

December 2017

Acknowledgements

I would first like to thank my advisor Dr. Ming-Ying Leung for the continuous support and encouragement in my study and thesis. Her guidance and immense knowledge has always helped me to give my best effort and be successful. I would also like to thank the rest of my thesis committee: Dr. Charlotte M. Vines, Dr. Rachid Skouta, and Dr. Xiaogang Su, for their encouragement, insightful comments, and questions.

I would like to thank my mother, Monowara Begum and my brother, Maksudul Alam for always supporting me and encouraging me to achieve my goals. I thank My Husband, Richard E. Mitchell for his tremendous care and support, and always giving me importance for anything I needed, also my son, Austin Mitchell, who always helped me in every possible way to work harder and encouraging me with his sweet words.

I am blessed to have wonderful friends like Ifeanyi H. Nwigboji, Bethuel Khamala and Sharmin Abdullah for always being there for me, helping me and treating me as a family. I thank Osei Tweneboah, Julio H. Solis, M Masum Bhuiyan, Dr. Andrew Pownuk and Kamal Nyaupane for being magnificent friends and classmates.

In conclusion, my sincere thanks to Jon Mohl for always helping me with everything and encouraging me to do better with my work. I thank Anastasia Kellog and Kyle Long for the initial data collection and constructing the framework of the database, as well as Gerardo Cardenas for the help in utilizing the bioinformatics computing facilities.

Abstract

G protein-coupled receptors (GPCRs) constitute the largest group of membrane receptor proteins in eukaryotes. Due to their significant roles in many physiological processes such as vision, smell, and inflammation, GPCRs are the targets of many prescribed drugs. However, the functional and structural diversity of GPCRs has kept their prediction and classification based on amino acid sequence data as a challenging bioinformatics problem. As existing computational methods to predict and classify GPCRs are focused on mammalian (mostly human) data, the ultimate goal of our project is to establish an ensemble approach and implement a web-based software that can be used reliably on a wider range of organisms. As a first step, we have constructed a searchable MySQL database with experimentally confirmed GPCRs and non-GPCRs along with protein features for distinguishing them. This database currently contains 2887 GPCR and 1614 non-GPCR sequences collected from the UniProtKB/Swiss-Prot protein database, covering over 300 species including arthropods, fungi, nematode, etc. Each protein in the database is assigned a unique identification number and linked to information about its source organism, sequence lengths, and other features including amino acid and dipeptide composition. For the GPCRs, family classifications according to the popular GRAFS and IUPHAR systems are also included. This database will provide the training and testing data for subsequent steps in our ongoing work to evaluate existing computational tools, incorporate them into our ensemble, and apply them to identify potential GPCRs in several fly, mosquito, and tick species that are of biomedical or agricultural importance.

Table of Contents

List of Tables

List of Figures

Chapter 1: Introduction

G-protein coupled receptors (GPCR) are the most vast and diverse group of membrane receptors in human genome. As the name implies, each GPCR binds to a G protein, and each G-protein binds the guanine nucleotides, guanosine diphosphate (GDP) and guanosine triphosphate (GTP). Each GPCR protein has a similar structure consisting a N-terminus, cytoplasmic C-terminus along with seven hydrophobic transmembrane (7TM) domains which are linked through three intracellular and three extracellular loops. There are approximately 800 [O'Hayre et al., 2013] distinct types of GPCRs in the human genome. Among these, there are over 140 orphan GPCRs [Stockert & Devi, 2015] whose functions are unknown and the rest of the GPCRs can be targeted by modern medicinal drugs. That is why predicting the GPCRs and classifying them to understand the functionality has a very high significance in the modern medical field.

Figure 1.1: G-protein-coupled receptor without ligand [C. Vines, Personal Communication, December 21, 2017]

1.1 GPCRs and their biomedical significance

GPCRs have an immense importance in several physiological processes and are targeted by many therapeutic developments due to their significant role in intracellular signaling and clinical importance in many diseases, especially cancer [Jo & Jung, 2016]. They can effect tumorigenesis and the growth of a tumor [Liu et al., 2016]. As GPCRs are involved in different but wide range of physiological processes including vision, taste, smell, inflammation, cell

recognition, pheromone signaling and many more, these transmembrane receptors are being studied for predicting the proteins and understanding their functions.

1.2 Prediction and Classification of GPCRs

To identify a GPCR and then classify, it is important to know the structure, the ligand binding sites and relative binding affinities. Classifications can be based on several features, such as protein sequence homology or functional similarity [Horn et al., 2003], vertebrates [Schiöth & Fredriksson, 2005] or vertebrates and invertebrates both [Alexander et al., 2015]. Horn et al. (2003), divides GPCRs into six major classes, named Class A (rhodopsin-like) [Alexander et al., 2015], Class B (secretin receptor family) [Poyner & Hay, 2012], Class C (metabotropic glutamate) [Palczewski et al., 2000], Class D (fungal mating pheromone receptors) [Eilers et al., 2005s], Class E (cyclic AMP receptors) [Raisley et al., 2004] and Class F (frizzled/smoothened) [(Gentry et al., 2015)]. Among these classes, Class D and E are unique to invertebrates. GRAFS [Schiöth & Fredriksson, 2005] is a classification system, which divides vertebrates GPCRs into five major classes named as Glutamate [Chaudhari et al., 2000], Rhodopsin-like [Zamanian et al., 2011], Adhesion [Langenhan et al., 2013], Frizzled/Taste2 [Krishnan et al., 2012], and Secretin family [Schiöth & Fredriksson, 2005]. The classes can be furthermore divided into sub-families, sub-sub-families and subtypes. It is important to study all the levels of classification because of the functional diversity of the GPCRs.

Figure 1.2: G-protein-coupled receptor with ligand [C. Vines, Personal Communication, December 21, 2017]

CLASS A

Class A is also known as Rhodopsin-like receptors which are the largest family of G-protein coupled receptors. The ligands for these receptors comprise of a large number of small molecules, peptides, hormones and neurotransmitters, while the receptors include olfactory receptors, taste receptors and five pheromone receptors [Alexander et al., 2015]. Rhodopsin-like receptors are studied in a wide range [Eilers et al., 2005] due to their major involvement in presecribed drugs [Venkatakrishnan et al., 2016].

CLASS B

Class B are called Secretin receptors which are known for controlling peptide hormones from the glucagon hormone family[Alexander et al., 2015]. Each member of this class has unique properties and plays an important role in metabolic diseases and physiology [Poyner & Hay, 2012].

CLASS C

This family is called the Glutamate family that consists of the metabotropic glutamate receptors along with several other receptors such as calcium-sensing receptor, taste receptors, and calcium-sensing receptors which play a significant role in several physiological processes such as calcium homeostasis, taste and synaptic transmission. The unique feature of this family is, it has a large extracellular N-terminus domain which mediates the homodimerization or heterodimerization [Wu et al., 2014].

CLASS D

This is a distinct type of family of GPCR which is also known as fungal mating pheromone receptors. This family comprises of a diverse group of receptors and the sequence similarity between the subfamilies is very low. The small peptide ligands binds to the extracellular loops and at the ends of transmembrane helices [Eilers et al., 2005].

CLASS E

Class E is known as the the cyclic adenosine monophosphate (cAMP) receptor family of GPCRs, which play a significant role in regulating the development related regulation genes. cAMP is used as intracellular signal transduction and acts as a second messenger for triggering the physiological changes [Raisley et al., 2004].

CLASS F

This class contains frizzled and smoothened proteins, which is why this family is also known as Frizzled/Smoothened family. The frizzled proteins are regulated by lipoglycoproteins of the WNT family and Smoothened family are indirectly regulated by the Hedgehog family [Alexander et al., 2015].

ADHESION FAMILY

This family belongs to the GPCR family classified by the GRAFS [Schiöth & Fredriksson, 2005] system for classification. Phylogenetically this family has close relationship with the class B (Secretin receptors) family but the receptors have high conservation, larger extracellular N-terminus, cell to cell and cell to matrix adhesion along with unique autocatalytic process which differs from the class B receptors [Hamann et al., 2015].

1.3 Objectives of the research

There are several existing computational methods for GPCR prediction and classification [Pearson & Lipman, 1988], [Altschul et al., 1990], [Elrod & Chou, 2002], [Karchin et al., 2002], [Qian et al., 2003], [Inoue et al., 2004], [Guo et al., 2006], [Iqbal et al., 2016], [Huang et al., 2004], [Gao & Wang, 2006], [Peng et al., 2010], [Davies et al., 2008], [Cobanoglu et al., 2011], [Li et al., 2010], [Sahin et al., 2014], [Guerrero et al., 2016], [Munoz et al., 2017] . These will be reviewed in Chapter 2. Generally, their reported prediction accuracies, are around the 80 – 90 % range. However, since these prediction algorithms are trained on GPCR sequences deposited in the database GPCRDB, where the vast majority of the GPCR data collected to date come from human or other mammals and used to target GPCRs for prescribed drugs, it is not clear at this

point how accurate the predictions would be when these computational methods are applied to the less studied species, such as arthropods.

The goal of my Ph.D. dissertation research is to design and implement a web-based ensemble of computational tools for prediction and classification of GPCRs. Expanding on the GPCR pipeline that my group has developed, this software will be publicly accessible via the Internet through a user-friendly graphical user interface. The ensemble will combine the strengths of different existing computational approaches, which include sequence and profile alignments, structural predictions, as well as machine learning and statistical data mining approaches. We will apply this ensemble tool to predict and classify GPCRs in several arthropod species that are of agricultural and biomedical importance. My specific aims are:

1. Develop a collection of datasets that contain a variety of known GPCRs and known non-GPCRs. These will include some well established datasets that have been used for benchmarking the performances of different approaches in the literature, and a new dataset that is developed within our group that contains experimentally confirmed GPCRs and non-GPCRs. A relational database will be created to form one integrated, searchable database.

Survey the existing computational methods for predicting and classifying GPCRs. The purpose of this survey is 4-fold:

(a) Compile a list of protein features (e.g., amino acid and dipeptide compositions, helix and loop lengths in GPCRs) that can be used for the prediction and classification.

(b) Understand the different algorithms (e.g., support vector machines, clustering) that are employed in the existing prediction and classification programs, and assess their predictive powers on the various datasets prepared in specific aim 1.

(c) Obtain access to the different programs. For each available program, we will find out whether it is an open source that can be downloaded and implemented locally. If it is open source, it will be downloaded and implemented as part of our ensemble. If it is web-based and requires manual submission, I will find suitable tools to automate the submission process.

(d) Integrate the outputs of the different programs to provide an overall summary of all the prediction results with an estimated confidence level.

2. Compare the prediction accuracies of different approaches using our dataset collections developed in Specific Aim 1 in order to identify the sequence features and prediction methods that to be included in the ensemble. In assessing the prediction accuracies, we will first make an overall evaluation using all the available data, and then repeat the evaluation only with sequence data collected for arthropods.

3. Collect the genomic data for 3 particular arthropods and their closely related species:

(a) The fruit fly *Drosophila melanogaster*, which is one of the most common model organisms in biological and medical science research. Compared to other arthropods, *Drosophila melanogaster* has the most information about its GPCRs collected in GPCRDB.

(b) The mosquito *Anopheles gambiae*, which is one of the most efficient malaria vectors known. Some closely related species are vectors for diseases like yellow fever and the West Nile virus.

(c) The southern cattle tick *Rhipicephalus microplus*, which is a vector of pathogens causing multiple health complication in cattle.

Carry out the prediction of which proteins are GPCRs in the above 3 species. Compile a web-accessible, publicly searchable database. At the same time, we will also collect all the experimentally confirmed GPCRs, as well as those previously predicted by others, to compare with our prediction.

As an intermediate milestone, I will report in this M.S. thesis the work accomplished towards specific aims 1, and propose how to approach the last part of specific aims 1, then specific aims 2 and 3. Chapter 2 is a review of the existing prediction methods. Chapter 3 explains of how the standard datasets for testing are obtained. Chapter 4 describes what our resulting datasets are like and what are the current list of features that are used in GPCR prediction and classification with their basic statistical characteristics. Chapter 5 delineates the

tasks to be completed in order to achieve all the specific aims in order, and how they will be approached in my dissertation research.

Chapter 2: Literature Review

This chapter reviews the different approaches and algorithms for the classification of GPCRs into various levels. The first section focuses on the computational tools which use machine learning and statistical approaches for the classification. The second section covers the combination of different methods followed by the third section which has the information about the programs and webservers which predicts and classifies GPCRs into various levels. In the fourth and last section, the comparison of programs and accuracies are discussed.

2.1 Computational tools for GPCR prediction and classification

Classification and prediction of GPCRs have been introduced by several approaches using various algorithms and computational methods. FASTA [Pearson & Lipman, 1988] and BLAST [Altschul et al., 1990] are the fundamental alignment search based tools for amino-acid sequences of proteins and the nucleotides of DNA sequences. Other than the search based methods there are statistical, machine learning and different methods used to predict and classify GPCRs into different families in which the protein sequence similarity is undetermined. Covariant discriminant algorithm [Elrod & Chou, 2002], support vector machines [Karchin et al., 2002], the hidden Markov model [Qian et al., 2003], binary topology pattern [Inoue et al., 2004], protein power spectrum using fast Fourier transform [Guo et al., 2006] and statistical encoding method [Iqbal et al., 2016], bagging classification tree [Huang et al., 2004], *k*-nearest neighbor (KNN) [Gao & Wang, 2006], and the principal component analysis[Peng et al., 2010] are the efficient methods to address the classification problem for higher levels. Combinations of several methods and tools has also been proposed to increase the accuracy for a larger dataset. Selective top-down classifier is also introduced by combining data mining and proteochemometrics

[Davies et al., 2008], support vector machines with maximum relevance minimum redundancy and genetic algorithm [Li et al., 2010]. However, in recent studies, different approaches have been developed where family specific motifs [Cobanoglu et al., 2011] or structural region lengths [Sahin et al., 2014] are used for prediction and classification of GPCRs. Other than human genomes, there has been research done for different organisms to reduce the risk of many diseases by using different tools such as TMHMM and GPCRpred for the transcriptome assembly of the cattle tick synganglion [Guerrero et al., 2016]. Furthermore encoding sequences from the transcriptome of the foreleg [Munoz et al., 2017] has been done using BLASTp, Pfam, GPCRpred, TMHMM, and PCA-GPCR for predicting GPCRs. With every algorithm, there are different sets of features which are studied to increase the efficiency of a certain method, such as amino acid composition, dipeptide composition, various physiochemical properties, cross validation and so on.

2.1.1 Support vector machine

Support vector machines (SVM) are known as the class of statistical learning algorithms that is mostly used for classification problem. Each data item is plotted in n-dimensional space where n is the number of features with the value of each feature being the value of a particular coordinate. Then the classification is performed by determining the hyper-plane that differentiates between the classes. SVM is very efficient but computationally expensive for classifying GPCRs into higher levels.

2.1.2 Covariant-discriminant algorithm

The covariant-discriminant algorithm introduced by Chou and Elrod (1999) is used to make an analysis of the correlation between the types of G-protein coupled receptors and their

amino acid composition. According to the GPCRDB [Horn et al., 1998] the rhodopsin-like amine GPCRs can be classified into six receptors namely acetylcholine, adrenoceptor, dopamine, histamine, serotonin and octopamine. For the training dataset histamine and octopamine receptors were removed as they are very low in quantity. Therefore, the study has been done for the rest of the four types of receptors which gives a 100% success rate on the re-substitution test. This method is efficient if a good training dataset can be established. The overall accuracy using the Jackknife rate for the dataset of 167 GPCRs is 83.23%.

2.1.3 Binary topology pattern

As GPCRs have highly divergent families, Binary Topology Pattern (BTP) [Inoue et al., 2004] is an efficient method for classification and identification with a good accuracy. Inoue et al., described it as a stepwise method where the first step works with three unified functional groups, (i) Class A and Non-GPCR, (ii) Class B + Class C and (iii) Frizzled/Smoothened, with a certain threshold value assigned. In the next step, it works with classifying Class B and Class C, and in step three, Class C is divided into three functional groups followed by Step four, five and six determining the rest of the functional groups along with the identification of the mammalian-type GPCRs. The accuracy is 100% for three groups, and four groups with more than 80%.

2.1.4 Bagging classification tree

Using tree based algorithm for prediction and classification is very effective and is used to address several classification problems. Huang et al. (2004) describes a bagging classification tree for classifying GPCRs into sub-families and sub-sub-families based on the amino-acid composition. Here it uses the C4.5 algorithm which is used to generate a decision tree. In total

4395 sequences were classified into sub-families and 4036 sequences were classified into sub-sub-families with an accuracy of 91.1% and 82.4% respectively.

2.1.5 K-nearest neighbor

Gao (2006) has introduced a nearest neighbor method to classify GPCRs from non-GPCRs and they further classified into four levels. The classification was done based on the amino acid composition and dipeptide composition of proteins. The dataset consists of 1406 GPCRs for classifying into six families and 1406 globular proteins where the accuracy is measured using the jackknife test and Matthew's correlation coefficient value. The accuracy is improved by using dipeptide composition for predicting GPCRs from the globular protein than using only amino acid composition. For further classification into six families, both amino acid and dipeptide composition have been used. Comparison of accuracy has been made with other existing methods, and it is observed that the accuracy of the existing method is better than the SVM-based method [Karchin et al., 2002] and covariant discriminant algorithm [Elrod & Chou, 2002]. There are no parameters to be determined for the nearest neighbor algorithm where amino acid and dipeptide composition have been used, this improved the simplicity in classifying GPCRs into four levels.

2.1.6 Principal component analysis (PCA)

Peng et al. (2010) proposed a method called PCA-GPCR for predicting and classifying GPCRs into family, sub-family, sub-sub-family, subtype from a large dataset. The study was done for 1497 sequence derived features which was further reduced into 32-dimensional feature vectors. In first level, the algorithm identifies whether the sequence protein is a GPCR or non-

GPCR, then for the GPCRs it is classified into four levels by using the intimate sorting algorithm. The accuracy for the prediction and classification is overall very high.

2.1.7 Family specific motifs

A method has been proposed by Cobanglu et al. (2011) for Class A sub-family classification of GPCRs that uses sequence derived motifs based on the receptor-ligand interaction sites. Distinguishing Power Evaluation (DPE) technique was proposed by Cobanglu et al. (2011) for the classification and also had a discovery of key ligand interaction sites.

2.1.8 Structural regional lengths

Sahin et al. (2014) introduced a method GPCRsort where the length of the transmembrane helices and loop regions was studied for predicting GPCRs using seven feature vectors which resulted in a fast prediction with a high accuracy of 97.3%. The study has been done on 38,525 protein sequences from GPCRDB and TMHMM is used for identifying the lengths of different regions of the GPCRs.

2.1.9 Statistical encoding method

Iqbal et al. (2016) worked in a statistical distance-based encoding method which works with the various distances of an amino acid in a sequence at different levels of decomposition to form a numeric feature vector. They worked on two different datasets to classify three families and sub-families of Class A. The overall accuracy is more than 94%.

2.2 Combination of tools

Studies have predicted and classified the GPCRs, where different types of tools are combined together to increase the efficiency for a large dataset or to lower the computational

cost. Mostly, it is observed that support vector machines with other tools are combined to get an effective output. This section describes briefly about the tool that has been worked on.

2.2.1 Support vector machine (SVM) with different approaches

Karchin et al. (2002) has used a simple nearest neighbor approach (BLAST), Hidden Markov Model (HMM) and support vector machines (SVMs) which is a group of statistical algorithms for recognizing superfamilies and the small subfamilies of GPCRs that bind the same ligand. The work is focused on comparing different methods with SVMs to observe which one is better computationally. For the classification of GPCRs, the primary sequence information is used which required the extension of the two-class problem to a k-class problem. Karchin et al. (2002) have used the simplest approach to multi-class SVMs by training k one-to-rest classifiers. SVM is computationally expensive but it has significantly less Minimum Error Point (MEP) than the other methods especially in the case for classifying subfamilies. It is also observed that the higher classification with good approximation can be achieved using SVMtree method. The future work is focused on classifying the subfamilies based on the suitable feature selection for the subfamilies along with the biological knowledge of each subfamily's transmembrane topology.

Yabuki et al. (2005) has described a system called GRIFFIN (G-Protein and Receptor Interaction Feature Finding Instrument) which uses Support Vector Machines and Hidden Markov Model to predict GPCR and G-Protein coupling selectivity along with a hierarchical SVM classifier including the feature vectors to predict Class A GPCRs. For the other type of families (Opsins, Ofactory subfamilies of Class A, Class B, Class C, frizzled and smoothened) HMM is used. As BLAST and FASTA uses sequence similarity for predicting the protein, yet it is not clear to predict GPCRs based on sequence similarity relationship. This system is unique as

it uses information from GPCR ligand information and GPCR sequence. SwissProt and TrEMBL databases are used as both ligand and sequence information are present. In total, they have used twenty-four features for ligands and GPCRs.

Another algorithm has been developed by [Liao, Ju, & Zou, 2016] which uses the features from SVM-Prot [Y. H. Li et al., 2016] and Random forest algorithm to identify GPCRs from non-GPCRs with an accuracy of 91.61%.

2.2.2 Fast Fourier transform with support vector machine

Guo et al. (2006) introduced a fast Fourier transform based support vector machine method to classify GPCRs and NRs based on the hydrophobicity of proteins. The three principal properties of hydrophobicity represented by hydrophobicity model, electron-ion interaction potential model and c-p-v model are used to transform the protein sequences into numerical sequences. Three hydrophobicity scales were selected for the optimization as hydrophobicity can vary due to different experimental conditions, different organic solvents and computing approaches. The dataset used for GPCRs is collected from GPCRDB containing 964 sequences for the final training dataset and for NRs, final training dataset of 465 sequences was observed which is collected from the NucleaRDB. For performance measurement Jackknife test is used as well as for prediction quality, accuracy, total accuracy and Matthew's correlation coefficient are evaluated. Higher accuracy is achieved with the hydrophobicity scale than c-p-v or electron-ion interaction potential model.

2.2.3 Feature selection with support vector machine

A three-layer classifier is proposed by Li et al. (2010) for GPCRs based on the combination of SVM with feature selection method. The method holds high accuracy for

classifying into superfamily, family and subfamily of GPCRs. For accuracy measurement Jackknife cross-validation test is used on two non-redundant datasets. Li et al. with 600 hundred features and then used the maximum relevance minimum redundancy to pre-evaluate features and used genetic algorithm is observed to find the optimized feature subset. For developing classification model support vector machine is coupled with the feature selection method. Higher accuracy is observed with the proposed method named GPCR-SVMFS than GPCR-CA and GPCRPred.

2.2.4 Genetic ensemble

Naveed and Khan (2012) introduces GPCR-MPredictor which predicts and classifies GPCRs into five levels (family, sub-family, sub-sub-family, subtype) including the prediction. It is an ensembled approach where *k*-nearest neighbor, support vector machine, probabilistic neural networs, J48, Adaboost and Naives Bayes classifiers have been used. Amino acid composition, pseudo amino acid composition and dipeptide composition these three features are used to predict and classify GPCRs. This ensembled approach has a higher accuracy than principal component analysis (PCA) method in all the five levels.

2.3 Databases

This section shows the several types of databases and web servers that are available for users so that if anyone wants to get information about the classification from a certain dataset, they can get it.

2.3.1 GPCRDB

GPCRDB contains data, diagram and tools and [Isberg et al., 2016] has the largest collection of receptor mutants and user friendly search option for the collection of crystal

structures. Users can create and collect the diagrams like snake-plot and box diagrams and phylogenetic trees for the illustration of receptor residues. There are many features that are unique and this database gives a lot of information as the sequence alignments considers helix bulges and constriction along with the amino acid conservation statistics with generic residue numbering. This database releases and updates data bi-monthly and currently they have 14,805 proteins, among which 414 are non-olfactory human proteins with a total of 3,547 organisms. It has search options for users such as receptors, signal proteins, ligands with pharmacophore generation and schematic aligments, experimental and mutation browser for crystal structures.

2.3.2 UniprotKB/SwissProt

This database [UniProt Consortium, 2008] contains the information about protein sequences along with the annotation data which is generally updated in every month. It gives an overall information about a protein. The main tools provided by this database are BLAST, sequence alignment, ID mapping and peptide search. It creates a collaborative search among four databases namely UniProtKB (protein knowledgebase), UniRef (Sequence clusters), UniParc (Sequence archive) and Proteomes. Currently this database has reviewed 555,594 sequences and 90,050,711 unreviewed sequences which are available through TrEMBL. UniProt is not limited to GPCR related proteins, but we can search for GPCR sequences and their information in this database.

2.4 Web-servers

There are several web servers available where single or multiple sequences can be inserted by the user that can predict and classify the proteins and they are publicly-available servers. The following section shows the web-servers which are free for users to use.

2.4.1 GPCRpred

GPCRpred [Bhasin & Raghava, 2004] is an SVM based method for the prediction of proteins in families and subfamilies level based on the dipeptide composition. As the webserver uses an old classification system, the family level classification is different than the one we have now. In the webserver there are three input options available for the query sequence such as inserting the standard format (PIR/FASTA/EMBL etc.) or the amino acid single letter code format or upload a file of the sequence.

The output page provides the information of name, sequence, length, date of prediction, the prediction approach (protein feature) used to predict and classify the query sequence. It is observed that the webserver works with one query sequence at a time. Following shows the output page for the query sequence that has been used for the demonstration purpose.

Figure 2.1: GPCRpred webserver with a query sequence

Figure 2.2: GPCRpred output page for the query sequence

2.4.2 GPCR-CA

GPCR-CA [Xiao et al., 2009; Chou, 2001; Chou & Elrod, 1999] predicts and classifies GPCRs into families using a cellular automaton image approach. GPCR-CA first predicts the protein to be a GPCR or non-GPCR, if it is a non-GPCR then the output page result shows that the query sequence is a non-GPCR otherwise for a predicted GPCR, it automates the process to classify the query sequence based on six main functional groups such as Rhodopsin-like, Secretin-like, Metabotrophic/Glutamate/Pheromone, Fungal pheromone, cAMP receptor and Frizzled/smoothened family.

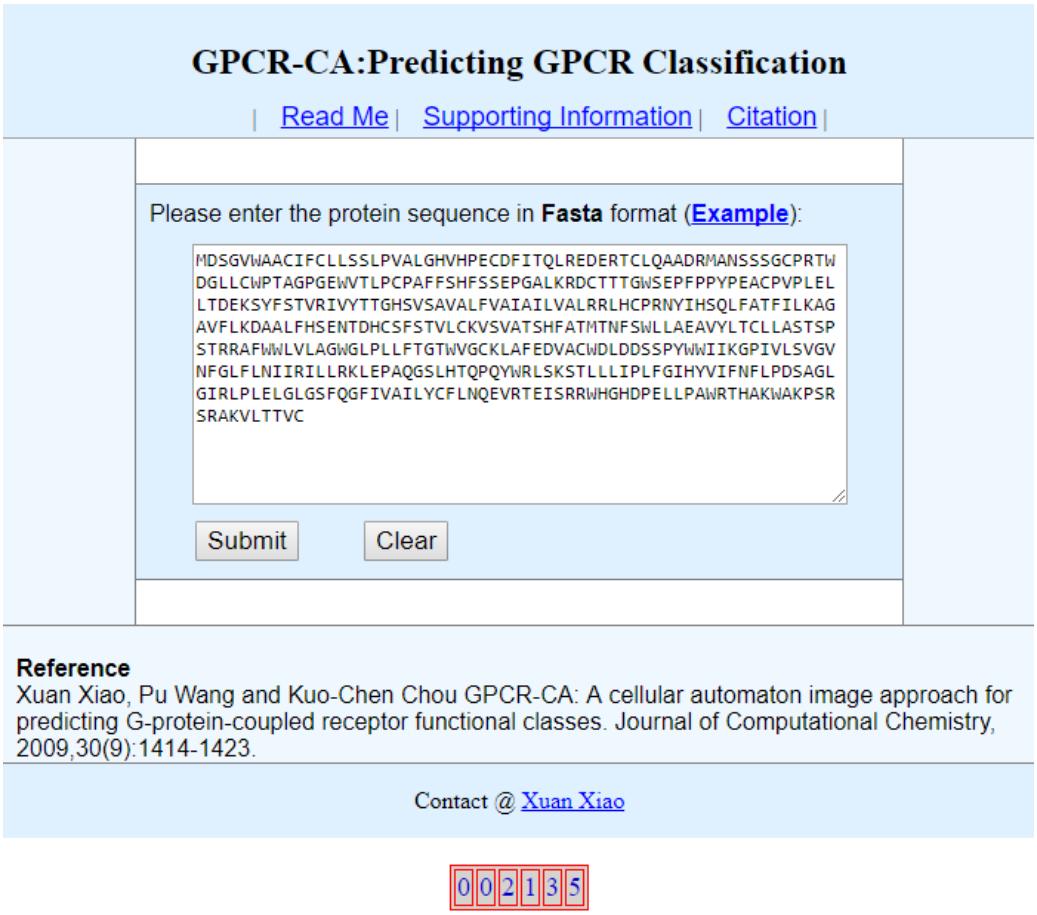


Figure 2.3: GPCR-CA webserver with a query sequence

The output for a query sequence is the length and the family level classification of a protein.

Following is the output result for a query sequence.

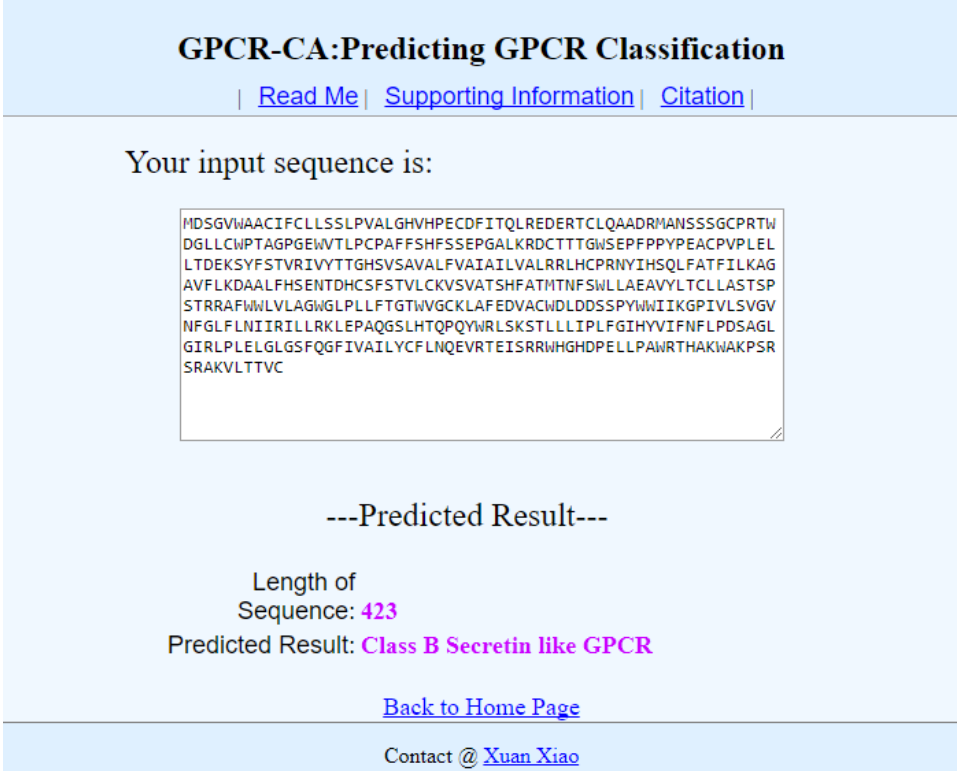


Figure 2.4: GPCR-CA output result for a query sequence

2.4.3 PRED-GPCR

This system is based on a probabilistic method [Papasaikas et al., 2004] which uses family specific profile HMMs to determine the classification of GPCRs into families for a given query sequence. The output of the query sequence results in a ranked list of the profile HMM, E-value cutoff, family, P-values, E-values and the number of profiles matched for each family.

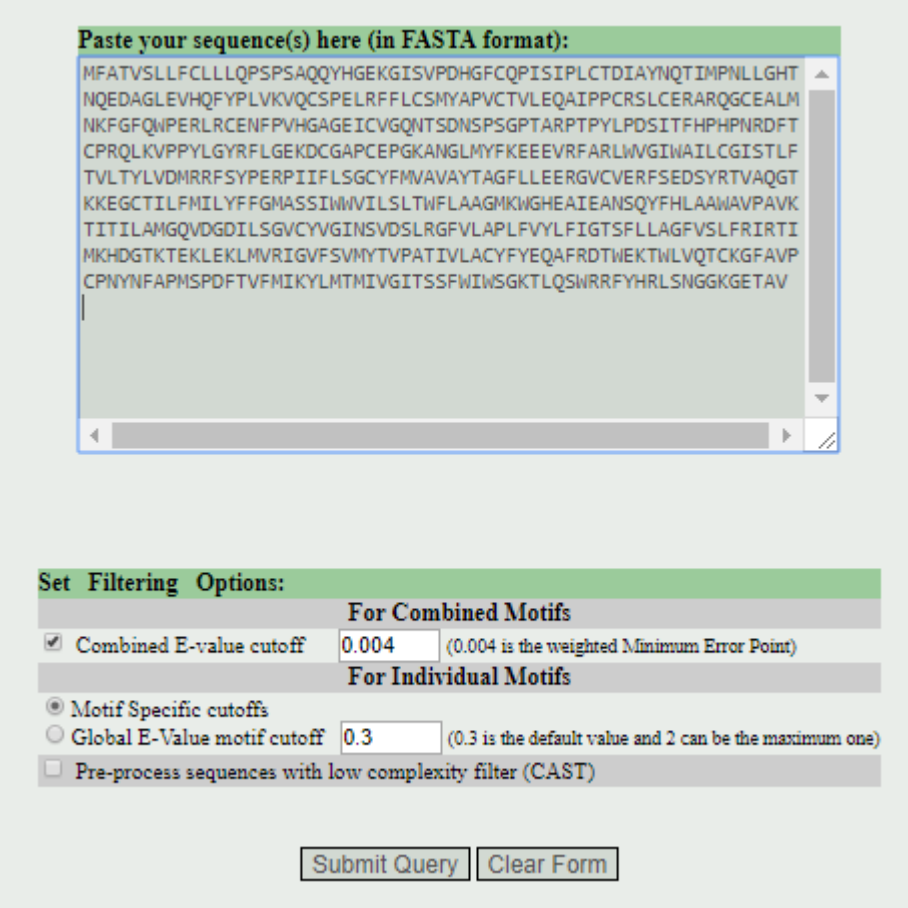
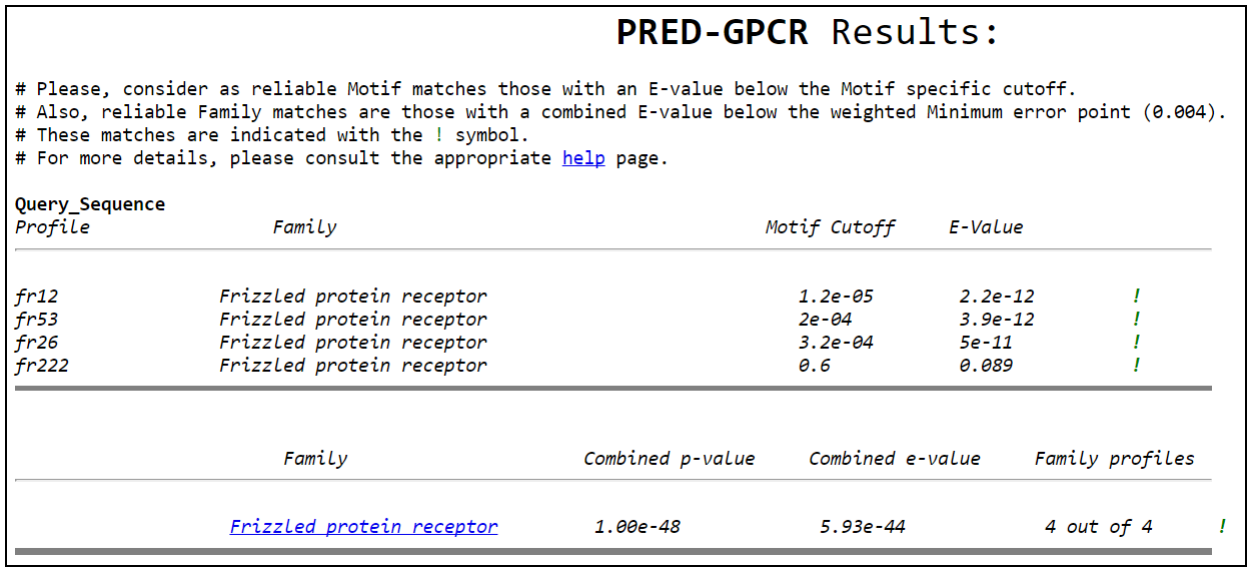


Figure 2.5: PRED-GPCR webserver with a query sequence

After submitting the query sequence, the output result page shows the motif cutoff, e-value, combined p-value, combined e-value and the protein profile information. If a non-GPCR sequence is used as a query, then the result page is blank.



PRED-GPCR Results:

Please, consider as reliable Motif matches those with an E-value below the Motif specific cutoff.
Also, reliable Family matches are those with a combined E-value below the weighted Minimum error point (0.004).
These matches are indicated with the ! symbol.
For more details, please consult the appropriate help page.

Query_Sequence

Profile	Family	Motif Cutoff	E-Value	
fr12	Frizzled protein receptor	1.2e-05	2.2e-12	!
fr53	Frizzled protein receptor	2e-04	3.9e-12	!
fr26	Frizzled protein receptor	3.2e-04	5e-11	!
fr222	Frizzled protein receptor	0.6	0.089	!

Family	Combined p-value	Combined e-value	Family profiles	
Frizzled protein receptor	1.00e-48	5.93e-44	4 out of 4	!

Figure 2.6: PRED-GPCR output page for the query sequence

2.4.4 SVM-Prot

This webserver [Li et al., 2016] which predicts functional families from the sequences of different types of proteins disregarding the sequence similarity. It provides an option to run a sample sequence available in the server also allows user to choose among three machine learning algorithms such as support vector machine, probabilistic neural network or *k*-nearest neighbor. User can provide multiple sequences to be analyzed, allows to access the result later by providing an email or a link. It also has a feature where user can perform a BLAST for the provided query sequence.

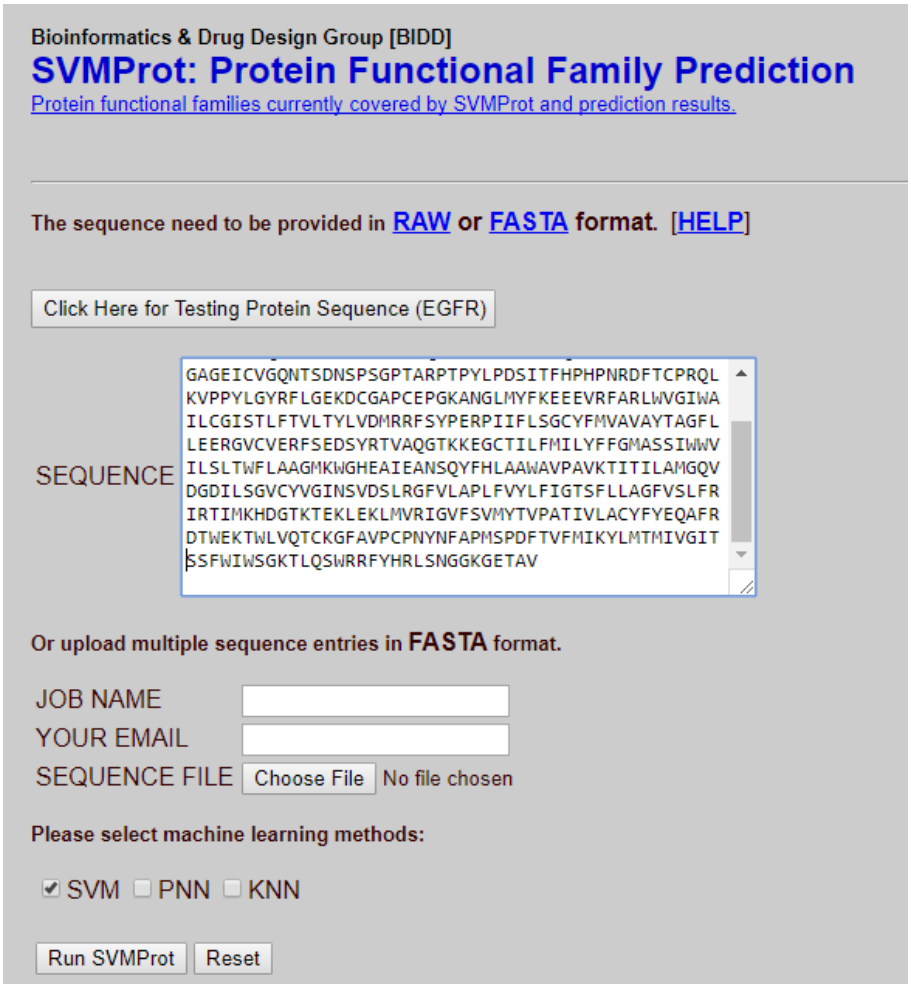
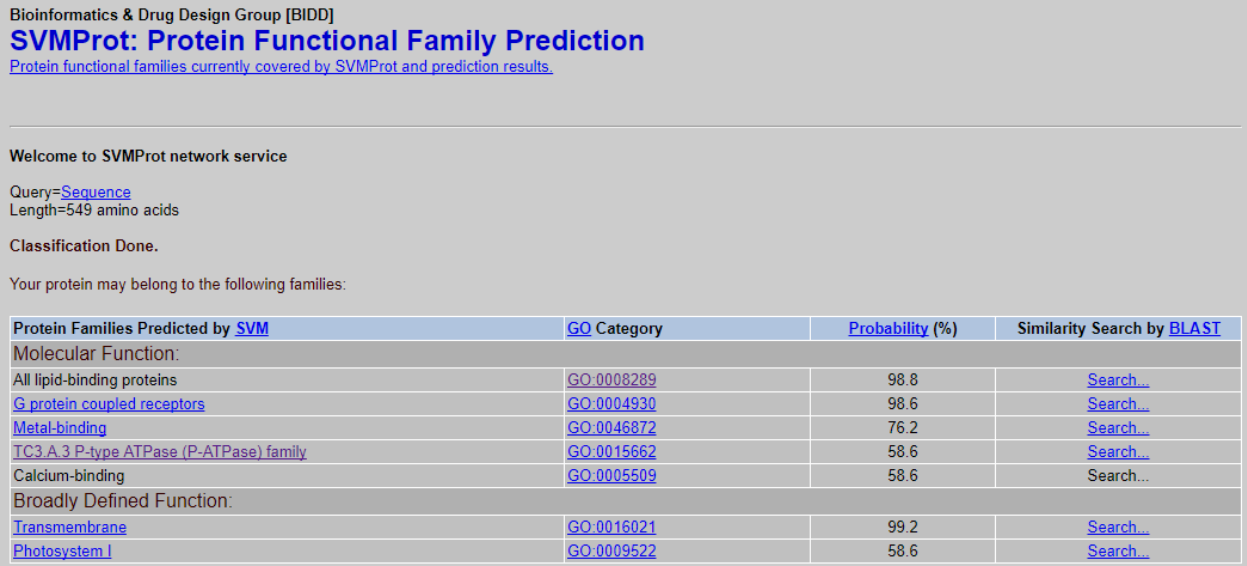


Figure 2.7: SVMProt webserver with a query sequence

The output page initially provides the waiting time and a link to check the results later for the user and when the analysis is done it provides user the result page that shows the probabilities of the protein families the sequence might belong to and the links to access those families with a BLAST option. The following is an example output page for a query sequence.



Bioinformatics & Drug Design Group [BIDD]

SVMProt: Protein Functional Family Prediction

Protein functional families currently covered by SVMProt and prediction results.

Welcome to SVMProt network service

Query=Sequence
Length=549 amino acids

Classification Done.

Your protein may belong to the following families:

Protein Families Predicted by SVM	GO Category	Probability (%)	Similarity Search by BLAST
Molecular Function:			
All lipid-binding proteins	GO:0008289	98.8	Search...
G protein coupled receptors	GO:0004930	98.6	Search...
Metal-binding	GO:0046872	76.2	Search...
TC3.A.3 P-type ATPase (P-ATPase) family	GO:0015662	58.6	Search...
Calcium-binding	GO:0005509	58.6	Search...
Broadly Defined Function:			
Transmembrane	GO:0016021	99.2	Search...
Photosystem I	GO:0009522	58.6	Search...

Figure 2.8: SVMProt output page for the query sequence

2.5 Comparison and accuracy

With different sets of algorithms and dataset, set of measurements have been taken to identify the accuracy for predicting and classifying GPCRs. Cross-validation and Jackknife test are the most commonly practiced methods along with sensitivity and specificity to measure the performance of a predictive model. The following table shows the information of the accuracy observed from different methods for a certain dataset.

Table 2.1: Summary of datasets, algorithms and accuracy measurements

Name	Dataset (sequences)	Prediction/Classification	Accuracy measurement	Accuracy
Covariant-discriminant [(Elrod & Chou, 2002)]	GPCRDB: 167	Correlation between GPCRs and amino acid composition	Re-substitution test Jackknife	Success rate 100%
Binary topology pattern [(Inoue et al., 2004)]	SwissPROT: 954 Mammalian GPCRs: 811 Non-GPCRs: 17	Ten groups	Sensitivity Specificity Success rate	100% for three groups >80% for four groups 65.8% for Non-GPCRs
Bagging classification	GPCRDB: 8431	Sub-families Sub-sub-families	Cross validation	91.1% for sub-family

tree [(Huang et al., 2004)]				82.4% for sub-sub-family
Fast Fourier transform [(Guo et al., 2006)]	GPCRDB: 946 (GPCRs)	(Class A to Class E)	Accuracy Total accuracy Matthew's correlation coefficient(MCC)	>94% accuracy except for Class C. >92% MCC for all the classes
GPCRTree [(Davies et al., 2008)]	GPRDB: 8222	Class Sub-family Sub-sub-family	GPCR servers	97% for class 84% for sub-family 75% for sub-sub-family
SVM & feature selection [(Z. Li et al., 2010)]	GPCR-CA: 730	Prediciton Class (A-E)	Jackknife Cross validation	98.22% for prediction 96.99% for class
SVM-Prot features & Random forest	Uniprot: 5026 GPCRs, 10386 non-GPCRs	Prediction	Cross validation Sensitivity Specificity Accuracy Matthew's correlation	91.61%
Principal component analysis (PCA) [(Peng et al., 2010)]	GPCRDB: 14026	Prediction Family Sub-family Sub-sub-family Subtype	Jackknife	99.5% for prediction 88.8% for family 80.47% for sub-family 80.3% for sub-sub-family 92.34% for subtype
Family specific motifs [(Cobanoglu et al., 2011)]	GPCRDB: 4889	Class A subfamilies (Amine, Peptide, Rhodopsin, Prostanoid, Olfactory)	Cross validation	90.7% overall
GPCR-MPredictor [(Naveed & Khan, 2012)]	GPCRDB and PCA: 14026	Prediction Family Sub-family Sub-sub-family Subtype	Jackknife test Sensitivity Specificity Matthew's correlation F-measures	99.75% for prediction 92.45% for family 87.80% for sub-family

				83.57% for sub-sub-family 96.17% for subtype
Structural region lengths [(Sahin et al., 2014)]	GPCRDB: 38525	Prediction And GPCRDB classifications	Cross validation	97.3%
Statistical encoding method [(Iqbal et al., 2016)]	GPCRDB: 2925	Class (A-C) Class A families: Dopamine, Serotonin, Chemokine	Cross validation	94-98%

Chapter 3: Materials and Methods

In this chapter we describe the construction of our local GPCR database, GPCR-PEnDB (GPCR Prediction Ensemble Database) which contains information about experimentally confirmed GPCRs and non-GPCRs. This is followed by a description of two other datasets that have been used in the literature to evaluate the performance of other GPCR prediction and classification algorithms. In the last section, we describe the features for predicting and classifying the GPCRs and how the features have been analyzed.

3.1 Construction of the GPCR-PEnDB database

The GPCR-PEnDB (GPCR Prediction Ensemble Database) database resides on apps02.bioinformatics.utep.edu which is a Dell PowerEdge R430 rack server that uses dual Intel Xeon E5-2620 processors (6 cores at 2.4GHz for each processor) and has two 16Gb DIMM memory modules. The server utilizes the CentOS 7 operating system, a Red Hat Enterprise Linux derivative, and is part of the Bioinformatics Network at UTEP. The physical server is located within the Research and Academic Data Center. The database is built in MySQL Version 14.14 Distribution 5.6.37, for Linux(x86_64) using EditLine wrapper. GPCR-PEnDB consists of GPCRs and non-GPCRs protein sequences, where GPCRs are downloaded from Swissprot (Uniprot) database. These were downloaded from www.uniprot.org using the advanced search options to filter by Uniprot's "protein family" function. The families searched were G-protein coupled receptor 1-5 family, which retrieves every family except for Frizzled. To get the Frizzled dataset the search is to be made using “G-protein coupled receptor fz smo family” and hence all the family datasets were downloaded separately and then merged. There are total 2887 protein sequences in this dataset. For non-GPCRs the sequences were downloaded from Swissprot where the sequences do not belong to the GPCR families, but the collection of sequences were higher than the GPCR dataset, therefore, a random sample of 3000 sequences was taken from the sequences and run through CDHIT with the threshold set at 0.50 which resulted in a more

diverse set of proteins as CDHIT clears out the sequences with greater than or equal to 50% sequence identity. Currently. the dataset consists of 1614 sequences of non-GPCRs.

GPCR-PEnDB is created by our group which initially had information on confirmed GPCRs from the Swissprot database, later it was populated with the non GPCRs. In the database the first table is named "PROTEINS" which has the GPCR and non GPCR IDs (PROTEINS_ID), name of the sequences (Sequence_name), protein sequences (Sequence), percentage of amino acids (A,C,…,Y,Others), Organism IDs (Organism_ID), IUPHAR IDs (IUPHAR_ID) ,GRAFS IDs (GRAFS_ID) and one column for distinguishing GPCRs from non-GPCRs (Protein_type). The primary key of that table is the PROTEIN_ID and the foreign keys are Organism_ID, IUPHAR_ID and GRAFS_ID which are used in the tables named "Organism", "IUPHAR" and "GRAFS" respectively. I have worked on the amino acid percentages for GPCRs (Appendix A) and then started populating (Appendix F) the database with the confirmed Non-GPCRs information in the existing tables.

The "Organism" table (Appendix F) initially had two columns containing the IDs and the scientific name and later it has been revised, where it has separated columns for generated organism IDs, genus, species, strain, serotype, common name and counts for each organism using Python (Appendix D). Also, for the counts of amino acids, dipeptides a table is generated called "AA_Dipeptide" where the primary key is the protein IDs.

The "TMHMM_Length" table (Appendix F) contains the lengths for different regions of a protein such as N-terminus (N_term), C-terminus (C_term), seven helices (TM1, TM2, TM3, TM4, TM5, TM6 and TM7), three inside loops (IL1, IL2 and IL3) and three outside loops (OL1, OL2 and OL3) are included which have seven transmembrane helices predicted by the TMHMM prediction tool and the data is generated using Python (Appendix D).

3.2 Collection of other datasets

To understand and survey the existing computational tools we have worked on collecting publicly available datasets that have been used for assessing different methods. There are two

datasets that have been collected and saved in our data collection drive. One contains the protein names and accuracies for the prediction and multi-level classifications which have been used by Peng et al., (2010) and Naveed and Khan, (2012). The second one is collected from GPCRsort [Sahin et al., 2014] which contains the prediction and probabilities of the family level classification of the proteins. Both of the datasets are collected and based on the classification system available on GPCRDB [Horn et al., 1998].

3.3 Feature collections for GPCRs and Non-GPCRs

In order to predict and classify GPCRs into different levels, there are different sets of features used by different algorithms to measure the accuracies of the methods as well as the efficacy of the features. Therefore, it is important to make a collection of the features along with the datasets for better observation and understanding. The objective here is to identify distinctive features used in different studies and later generate them for our dataset using a suitable programming language. The first protein feature that has been worked on is the percentage of 20 amino acids for each protein sequences available in the dataset. Python programming language has been used to generate the "CSV" (comma separated values) file (Appendix A) that contains the sequence IDs and percentages. The second protein feature is the dipeptide composition (Appendix C) which has been used in several studies based on different algorithms where, for every sequence there are total 400 dipeptide compositions.

The third protein feature selection covers the GPCRs with seven transmembrane helices that contains the length of N-term, three inside loop, three outside loop and C-term which has been calculated using the prediction TMHMM 2.0 tool using Python (Appendix E), which gives the transmembrane helices (TMhelix), inside and outside loop lengths including the total number of transmembrane helices. TMHMM 2.0 is used in our bioinformatics test-server which runs with the operating system CENTOS07. In this case, the sequences that have been predicted to be composed of seven transmembrane helices are used for calculating the lengths of different regions and generating plots to observe the data. As TMHMM is only a prediction tool where

errors may be possible, the predicted protein sequences which have close to seven transmembrane helices (6 or 8) are also kept in a separate FASTA file so that we can examine the sequences later. In total, there are 2190 out of 2887 GPCR protein sequences with seven transmembrane helices predicted by TMHMM available in the database.

Chapter 4: Results

In this chapter the overall result that has been achieved by constructing the database, the description, entity relationship diagram of the database and the results obtained by analyzing the features have been discussed.

4.1 The GPCR-PEnDB database

In GPCR-PEnDB database there are eight tables namely PROTEINS, Organism, AA_Dipeptide, TMHMM_Length, IUPHAR, GRAFS, GPCR_GeneOntology and GPCR_PDB using the MySQL language. The main focus here is on the different protein features, classification levels and types of organisms. The PROTEINS table contains the sequence ids, protein names, entry names, length of the sequences, the amino acid percentages and the type of protein for all the sequences. In this table IDs have been assigned to the protein sequences based on the GRAFS and IUPHAR system along with the IDs assigned for each different organism type which allows them to connect with the GRAFS, IUPHAR and Organism tables respectively using foreign keys for the IDs.

In the Organism table (Appendix F), all the entities have their scientific names, strain, serotype and common names along with an identification number. An additional column named "Frequency" has the counts of how many sequences are available in the dataset for each type of organism. There are total 1101 Organism IDs present in the database which covers all the 4501 protein sequences in the database. The purpose of this structure is to make it easier for studying different organisms and distinguish them based on their taxonomy or certain characteristics type. User will be able to narrow down the search for studying the features if it focuses on a set of organisms.

The AA_Dipeptide table (Appendix F) contains 422 columns, the first one is the name of the protein, 21 columns for 20 types of amino acids and the undetermined amino acid (Others) that has been found in the sequences. The rest of the 400 columns are for the counts of each dipeptide combination in every sequence. So, in general this table contains the most basic protein

features that have been used to predict GPCRs from non-GPCRs and later classified to different family levels using different algorithms.

The THMHMM_Length table (Appendix F) contains the length of different regions for a GPCR protein found by the TMHMM2.0 prediction tool which helps us to look at the diversity or the common characteristics of the proteins. This table contains only those GPCRs which have been predicted to have 7 transmembrane helices and currently there are data available for 2190 sequences.

The GRAFS and the IUPHAR table have the same structure with having two columns, first column contains the IDs and the second column has the classification names for the families. The following shows the entity relationship diagram of the database with the tables that have been worked on.

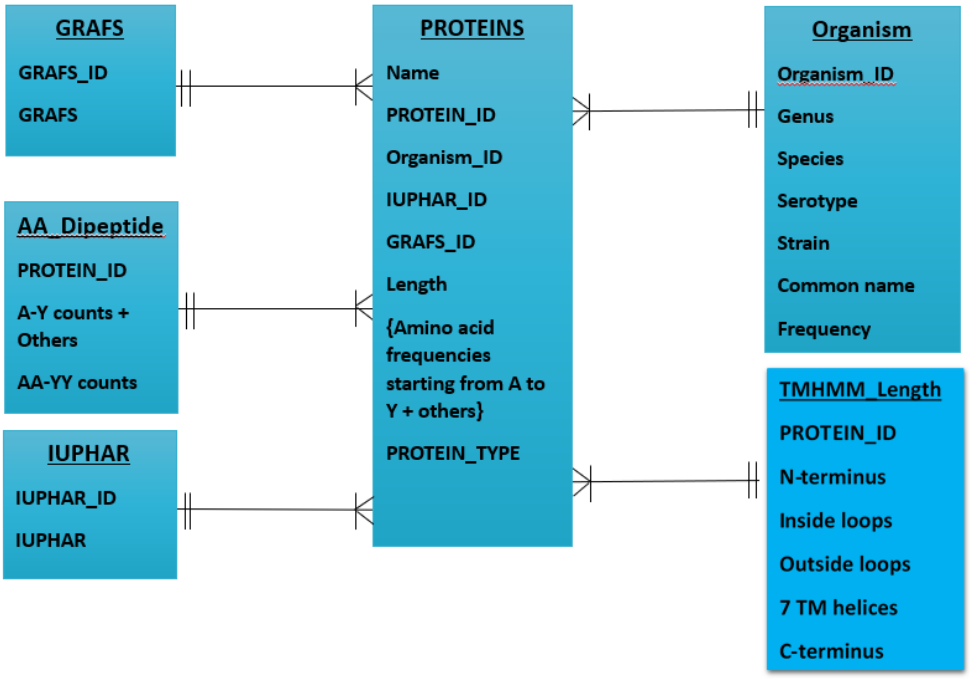


Figure 4.1: GPCR-PEnDB entity relationship diagram

4.2 Results of protein features and datasets analyzed

This section is divided into two parts, the first section focuses on the results of the features that we have collected from analyzing the data from our database. The second section has a brief description on the datasets that we have collected and analyzed.

4.2.1 Results of features

For the prediction of GPCRs and Non-GPCRs from a total of 4501 protein sequences, the logistic regression method gives a total of 93.33% accuracy with the percentages of amino acids used as the feature selection.

The SwissProt dataset which contains all the GPCRs, has been used to analyze the length of different regions such as N-terminus, transmembrane helices, outside loop, inside loop and C-terminus found by the prediction tool TMHMM 2.0. The program is written in python language to generate the lists of the different region lengths and then R language (Appendix G) is used to generate the boxplots and histograms along with the five number summaries. The following is the boxplot for N-terminus, sum of seven transmembrane helices, sum of the three outside loops, sum of the three inside loops and C-terminus respective.

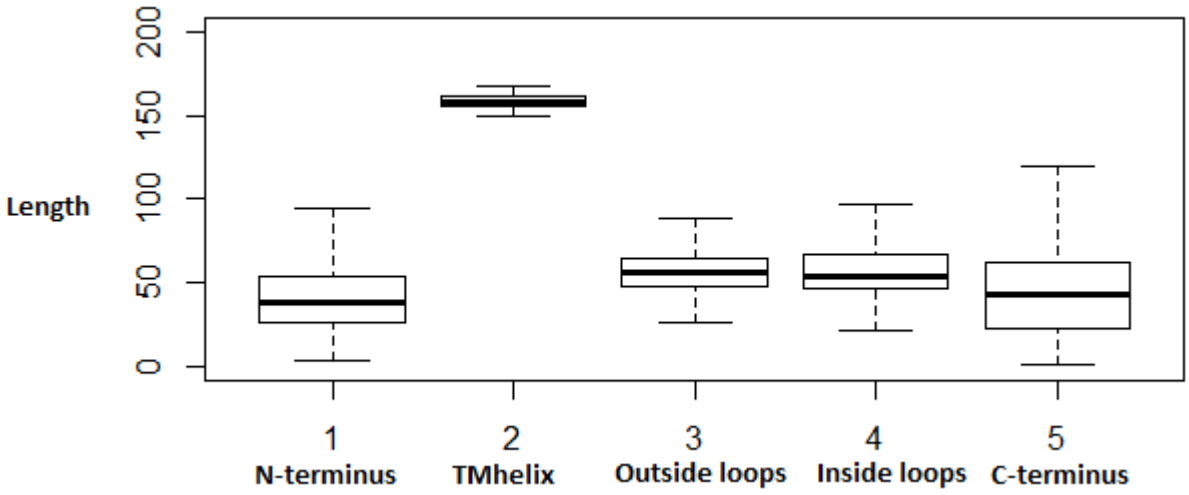


Figure 4.2: Boxplot of the protein lengths

The following histograms are generated by summing the different regions (helices, outside and inside loops). Which tells us that there are outliers in the dataset and information of the protein can be found using the search-based database.

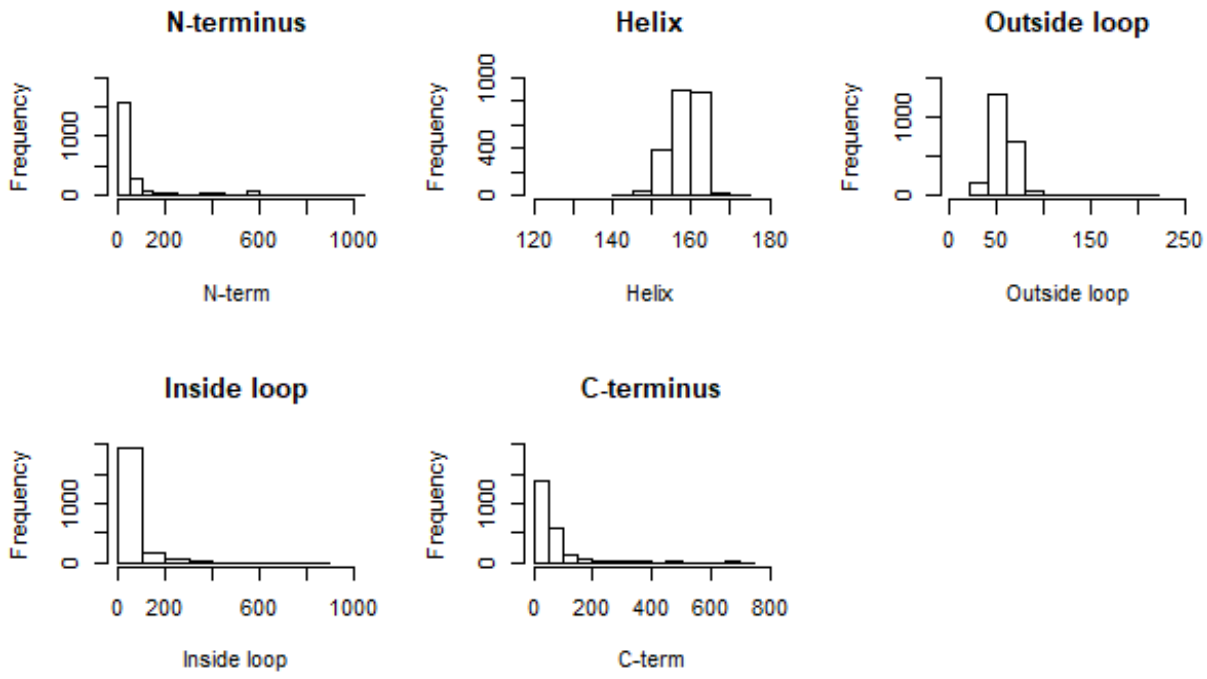


Figure 4.3: Histograms of the protein lengths

To identify the protein features for the protein sequences which have a N-terminus greater than 1000 a search has been made in the database to get a list of the protein names which has a large N-terminus and the following result is obtained.

```
mysql> select PROTEINS.Name,PROTEINS.PROTEIN_ID,TMHMM_Length.N_term from PROTEINS, TMHMM_Length WHERE PROTEINS.PROTEIN_ID = TMHMM_Length.PROTEIN_ID AND N_term>1000;
+-------------+------------+--------+
| Name        | PROTEIN_ID | N_term |
+-------------+------------+--------+
| AGRG4_MOUSE | B7ZCC9     | 2693   |
| AGRF5_MOUSE | G5E8Q8     | 1015   |
| CELR1_MOUSE | O35161     | 2485   |
| GPR98_DANRE | Q6JAN0     | 5803   |
| AGRF5_HUMAN | Q8IZF2     | 1013   |
| AGRG4_HUMAN | Q8IZF6     | 2743   |
| GPR98_HUMAN | Q8WXG9     | 5907   |
| CELR2_HUMAN | Q9HCU4     | 2378   |
| CELR1_HUMAN | Q9NYQ6     | 2470   |
| CELR2_RAT   | Q9QYP2     | 1603   |
| CELR2_MOUSE | Q9R0M0     | 2379   |
| AGRF5_RAT   | Q9WVT0     | 1016   |
+-------------+------------+--------+
12 rows in set (0.00 sec)

mysql> select PROTEINS.Name,PROTEINS.PROTEIN_ID,TMHMM_Length.N_term from PROTEINS, TMHMM_Length WHERE PROTEINS.PROTEIN_ID = TMHMM_Length.PROTEIN_ID AND N_term>5000;
+-------------+------------+--------+
| Name        | PROTEIN_ID | N_term |
+-------------+------------+--------+
| GPR98_DANRE | Q6JAN0     | 5803   |
| GPR98_HUMAN | Q8WXG9     | 5907   |
+-------------+------------+--------+
2 rows in set (0.00 sec)
```

Figure 4.4: Database search for N-term length larger than 1000 and then 5000

The next step can be done to identify the Protein IDs and the names of the protein sequences with the rest of the features from the TMHMM_Length and Organism tables.

```
mysql> select PROTEINS.Name,Organism_v2.Common_name FROM PROTEINS,Organism_v2 WHERE PROTEINS.PROTEIN_ID = 'Q8WXG9' AND PROTEINS.Organism_ID = Organism_v2.Organism_ID;
+-------------+-------------+
| Name        | Common_name |
+-------------+-------------+
| GPR98_HUMAN | Human       |
+-------------+-------------+
1 row in set (0.00 sec)

mysql> select PROTEINS.Name,Organism_v2.Common_name FROM PROTEINS,Organism_v2 WHERE PROTEINS.PROTEIN_ID = 'Q6JAN0' AND PROTEINS.Organism_ID = Organism_v2.Organism_ID;
+-------------+-------------+
| Name        | Common_name |
+-------------+-------------+
| GPR98_DANRE | Zebra fish  |
+-------------+-------------+
1 row in set (0.00 sec)
```

Figure 4.5: Search for the protein names by using the protein IDs

In the following query then lengths of different regions have been observed by looking for specific protein IDs.

```
mysql> select *from TMHMM_Length where PROTEIN_ID = 'Q6JAN0' OR PROTEIN_ID ='Q8WXG9';
+------------+--------+------+------+------+------+------+------+------+------+------+------+------+------+------+--------+--------+
| PROTEIN_ID | N_term | IL1  | IL2  | IL3  | TM1  | TM2  | TM3  | TM4  | TM5  | TM6  | TM7  | OL1  | OL2  | OL3  | C_term | Length |
+------------+--------+------+------+------+------+------+------+------+------+------+------+------+------+------+--------+--------+
| Q6JAN0     | 5803   | 8    | 11   | 25   | 23   | 23   | 23   | 23   | 23   | 23   | 23   | 14   | 23   | 4    | 150    | 6199   |
| Q8WXG9     | 5907   | 8    | 11   | 20   | 23   | 23   | 23   | 23   | 23   | 23   | 23   | 14   | 25   | 9    | 151    | 6306   |
+------------+--------+------+------+------+------+------+------+------+------+------+------+------+------+------+--------+--------+
2 rows in set (0.00 sec)
```

Figure 4.6: Lengths of different regions for given protein IDs

It is possible to make different searches based on different characteristics of the proteins using the database which gives us the advantage for an efficient characteristic based search. The following is the five number summaries of the different regional lengths for the predicted 2190 GPCR proteins.

Table 4.1: Five number summary of lengths

Name	Minimum	Lower-quartile	Median	Upper-quartile	Maximum
N-term	3	26	38	54	5907
Helix	144	156	158	161	172
Outside	26	48	56	64	205
Inside	21	47	54	67	874
C-term	1	23	43	62	718

4.2.2 Description of the other datasets

The dataset used for PCA-GPCR [Peng et al., 2010] and GPCR-MPredictor [Naveed & Khan, 2012] have used for 3178 protein sequences for predicting GPCRs (1589 proteins) and Non-GPCRs (1589 proteins). For classifying into multi-level they have used 1589, 4772, 4924 and 2741 for family, subfamily, sub-sub family and subtype levels respectively. There are counts given for the total number of sequences, correctly predicted sequences and the accuracy. For different family level classifications, they have used different identity levels such as for predicting GPCRs from non GPCRs and for family level classification the dataset has less than 40% identity with other sequences, for subfamily level it has less than 70% identity, for sub-subfamily it has less than 80% identity and for subtype it has less than 90% identity with other sequences. The name of families, subfamilies, sub-subfamilies and subtypes are collected from the GPCRDB[Horn et al., 1998] database and they have used a unique identification number system for naming the proteins, such as for sub-subfamily classification if the protein name is 001-001-003, it means sub-subfamily "Dopamine" is the subfamily of the *Subfamily* 001-001 which is "Amine" and subfamily of the family is "Class A Rhodopsin like".

Using the structural region lengths, Sahin et al., (2014) have used 38525 sequences from GPCRDB [Horn et al., 1998] database and classified them according to the GPCRDB database. The result is obtained by 10-fold cross- validation experiments and the supplementary file contains the protein IDs and actual family names, the predicted family and the probability distribution for different family levels.

Chapter 5: Future Work

5.1 Research goals

As stated in section 1.3, the final goal of my Ph.D. dissertation research is to design and implement an ensemble of computational tools that can be used to reliably predict and classify GPCRs for different organisms. To date, I have completed the major part of first specific aim of establishing several datasets, including the GPCR-PEnDB relational database, and two other datasets that have been used in the literature for evaluating some existing algorithms.

Currently, I am working on the last part of the first specific aim along with the second specific aim, which involves collecting the available GPCR prediction and classification programs that comprise the ensemble. This requires understanding the underlying algorithms, gathering the required protein features, implementing the programs' source code in our local computers, and making them accessible on our local GPCR website gpcr.utep.edu.

Completion of the work in specific aim 1 will allow the prediction accuracies of the different programs to be compared on our established datasets. It is important not only to understand how the methods are predicting and classifying the GPCRs but also assess the size of the datasets that they can handle, and evaluate their accuracies in different types of organisms other than human and other mammalian proteins. While these assessments are being conducted, I will also collect and organize the available protein sequence data from three arthropod species, namely the fruit fly *D. melanogaster*, the mosquito *A. gambiae*, and the cattle tick *R. microplus*. These organisms are chosen not only for their scientific, medical, and agricultural importance, but also because of their relevance to the research interest of local researchers at UTEP. I will then select the best ensemble of computational tools to conduct GPCR prediction and classification for the protein sequences of these species. The prediction results will be compared to other GPCRs from these arthropods recorded in the literature and compile a publicly searchable database module to be added to GPCR-PEnDB.

5.2 Proposed methods

The research goals stated above will be accomplished through the procedures described in this section.

5.2.1 Assembling existing computational tools (the last part of specific aim 1)

In the literature review process of this thesis, I have collected the information on all the existing GPCR prediction and classification programs and their underlying algorithm designs, along with the protein features used. In general, support vector machines have been found very effective but computationally expensive for this purpose but there are also other methods such as covariant discriminant method, principal component analysis, binary topology pattern and other statistical methods that has been seen to achieve good accuracies. Among these programs, I will identify the ones with publicly available source codes, implement them on the local computers in the bioinformatics network, and obtain a detailed understanding of the strengths and limitations of each algorithm.

5.2.2 Comparison of prediction accuracies (specific aim 2)

With the help of the bioinformatics technical staff, the computational tools assembled in specific aim 1 will be incorporated into our existing GPCR pipeline to facilitate the comparison of prediction results. From the sequence data compiled in specific aim 1, I will generate training and testing datasets for evaluation of the computational tools. It is important to ensure that the datasets cover a wide range of organisms. In addition to the overall prediction accuracy, the sensitivity, specificity, as well as positive and negative predictive values will be recorded. Not only the overall average accuracies of the tools will be compared against each other, but their performance on particular groups of organisms also need to be evaluated. For the purpose of our subsequent specific aims, special attention will be paid to the prediction accuracies for the arthropods. Based on these accuracy comparison results, we will choose the optimal ensemble that will be used to predict and classify GPCRs for our three arthropod species of interest.

5.2.3 Collection of sequence data for selected organisms and compilation of GPCR prediction results (specific aim 3)

The proteomes of *D. melanogaster* (21,976 proteins) and *A. gambiae* (13,515 proteins) can be downloaded from UniProt, but only 778 proteins of *R. microplus* are available in UniProt. However, a draft genome of it is published only in 2017 [Barrero et al., 2017]. With the genomic sequence data, we can at least predict the protein-encoding regions, translate them into protein sequences, and then try to predict which ones are GPCRs. This procedure, used on the *R. microplus* synganglion and Haller's organ transcriptome sequences, has been described by Guerrero et al., 2016 and Munoz et al., 2017. The program is available as part of the GPCR pipeline. I will utilize this to obtain the protein sequences of *R. microplus*.

The collected sequence data will be submitted to the optimal ensemble selected in specific aim 2. The predicted results will be compiled and compared with other confirmed or predicted GPCRs of the same organisms that have been reported in the literature. Noting that such information is quite scattered in publications from different journals, and finding all of them will take a good amount of time, we will start this process as soon as possible. I will combine this information with our own predicted results and construct additional tables that will be integrated to GPCR-PEnDB so that the information can be searched by researchers. As we expect that the GPCR information about these organisms would change over time as more of the proteins are confirmed to be GPCRs and non-GPCRs, these tables will be constructed in a way that future addition or changes can be processed conveniently.

5.3 Expected results and possible pitfalls

The main result of my work is the establishment of an ensemble approach of the studied algorithms, which is expected to be a robust tool for discovering possible new GPCR, which can then be further investigated by experimental scientists in the wet lab. During the development process of this ensemble approach, we will be able to determine the strengths and weaknesses of the various existing computation tools when used on datasets of different sizes, and sequences from different groups of organisms. Furthermore, the completion of the searchable GPCR-

PEnDB database that include information and making it accessible through a web-based user interface will provide a helpful resource for researchers who are developing treatments for mosquito and tick related diseases.

In the implementation of the ensemble approach, we anticipate that there may be issues related to the computational demands of the existing computational algorithms. Some of them, especially the ones based on support vector machines, require large amounts of computing time. While we can get around these problems by using multiple computers in the bioinformatics network for the computation required by my research, making the ensemble software publicly accessible open to the public through the internet may create a big computational burden beyond the capacity of current bioinformatics computing facilities. We may need to make arrangements with the UTEP Higher Performance Computing office to allow us to use their equipment. The logistics of such arrangements have to be worked out.

5.4 Timeline

The following table shows the tentative timeline for the research goals to be completed.

Table 5.1: Timeline for the completion of research goals

Date	Tasks to complete
January 2018 - May 2018	• Creating the web application of the database. • Study and collect information available for different algorithms. • Including Predicted GPCRs, non-GPCRs sequences in the database.
June 2018 - August 2018	• Collection of new protein features in the database. • Collection of the arthropods sequences from the UniProt and other public data banks.
September 2018 – December 2018	• Analyze the sequences using the algorithms and accumulate the results. • Compare prediction accuracies of different algorithms.
January 2019 – May 2019	• Validating the results obtained from the analysis and include the results in the dissertation drafts.

June 2019 – August 2019	• Finalize Ph.D. Dissertation and prepare for defense.

References

Alexander, S. P., Davenport, A. P., Kelly, E., Marrion, N., Peters, J. A., Benson, H. E., … Collaborators, C. (2015). The Concise Guide to PHARMACOLOGY 2015/16: G protein-coupled receptors; The Concise Guide to PHARMACOLOGY 2015/16: G protein-coupled receptors. *British Journal of Pharmacology*, *172*, 5744–5869. https://doi.org/10.1111/bph.13348/full

Altschul, S. F., Gish, W., Miller, W., Myers, E. W., & Lipman, D. J. (1990). Basic local alignment search tool. *Journal of Molecular Biology*, *215*(3), 403–410. https://doi.org/10.1016/S0022-2836(05)80360-2

Barrero, R. A., Guerrero, F. D., Black, M., McCooke, J., Chapman, B., Schilkey, F., … Bellgard, M. I. (2017). Gene-enriched draft genome of the cattle tick Rhipicephalus microplus : assembly by the hybrid Pacific Biosciences/Illumina approach enabled analysis of the highly repetitive genome. *International Journal for Parasitology*, *47*(9), 569–583. https://doi.org/10.1016/j.ijpara.2017.03.007

Bhasin, M., & Raghava, G. P. S. (2004). GPCRpred: An SVM-based method for prediction of families and subfamilies of G-protein coupled receptors. *Nucleic Acids Research*, *32*(WEB SERVER ISS.), 383–389. https://doi.org/10.1093/nar/gkh416

Chaudhari, N., Landin, A. M., & Roper, S. D. (2000). A metabotropic glutamate receptor variant functions as a taste receptor. *Nature Neuroscience*, *3*(2), 113–119. https://doi.org/10.1038/72053

Chou, K.-C. (2001). Prediction of protein cellular attributes using pseudo-amino acid composition. *Proteins: Structure, Function, and Genetics*, *43*(3), 246–255. https://doi.org/10.1002/prot.1035

Chou, K.-C., & Elrod, D. W. (1999). Protein subcellular location prediction. *Protein Engineering, Design and Selection*, *12*(2), 107–118. https://doi.org/10.1093/protein/12.2.107

Cobanoglu, M. C., Saygin, Y., & Sezerman, U. (2011). Classification of GPCRs using family specific motifs. *IEEE/ACM Transactions on Computational Biology and Bioinformatics*, *8*(6), 1495–1508. https://doi.org/10.1109/TCBB.2010.101

Davies, M. N., Secker, A., Halling-Brown, M., Moss, D. S., Freitas, A. a, Timmis, J., … Flower, D. R. (2008). GPCRTree: online hierarchical classification of GPCR function. *BMC Research Notes*, *1*, 67. https://doi.org/10.1186/1756-0500-1-67

Eilers, M., Hornak, V., Smith, S. O., & Konopka, J. B. (2005). Comparison of class A and D G protein-coupled receptors: common features in structure and activation. *Biochemistry*, *44*(25), 8959–75. https://doi.org/10.1021/bi047316u

Elrod, D. W., & Chou, K. C. (2002). A study on the correlation of G-protein-coupled receptor types with amino acid composition. *Protein Eng*, *15*(9), 713–715.

Gao, Q. Bin, & Wang, Z. Z. (2006). Classification of G-protein coupled receptors at four levels. *Protein Engineering, Design and Selection*, *19*(11), 511–516. https://doi.org/10.1093/protein/gzl038

Gentry, P. R., Sexton, P. M., & Christopoulos, A. (2015). Novel Allosteric Modulators of G Protein-coupled Receptors. *The Journal of Biological Chemistry*, *290*(32), 19478–88. https://doi.org/10.1074/jbc.R115.662759

Guerrero, F. D., Kellogg, A., Ogrey, A. N., Heekin, A. M., Barrero, R., Bellgard, M. I., … Leung, M.-Y. (2016). Prediction of G protein-coupled receptor encoding sequences from the synganglion transcriptome of the cattle tick, Rhipicephalus microplus. *Ticks and Tick-Borne Diseases*, *7*(5), 670–677. https://doi.org/10.1016/j.ttbdis.2016.02.014

Guo, Y. Z., Li, M., Lu, M., Wen, Z., Wang, K., Li, G., & Wu, J. (2006). Classifying G protein-coupled receptors and nuclear receptors on the basis of protein power spectrum from fast Fourier transform. *Amino Acids*, *30*(4), 397–402. https://doi.org/10.1007/s00726-006-0332-z

Hamann, J., Aust, G., Arac, D., Engel, F. B., Formstone, C., Fredriksson, R., … Schioth, H. B. (2015). International Union of Basic and Clinical Pharmacology. XCIV. Adhesion G Protein-Coupled Receptors. *Pharmacological Reviews*, *67*(2), 338–367. https://doi.org/10.1124/pr.114.009647

Horn, F. (2003). GPCRDB information system for G protein-coupled receptors. *Nucleic Acids Research*, *31*(1), 294–297. https://doi.org/10.1093/nar/gkg103

Horn, F., Weare, J., Beukers, M. W., Hörsch, S., Bairoch, A., Chen, W., … Vriend, G. (1998). GPCRDB: an information system for G protein-coupled receptors. *Nucleic Acids Research*, *26*(1), 275–279. https://doi.org/10.1093/nar/26.1.275

Huang, Y., Cai, J., Ji, L., & Li, Y. (2004). Classifying G-protein coupled receptors with bagging classification tree. *Computational Biology and Chemistry*, *28*(4), 275–280. https://doi.org/10.1016/j.compbiolchem.2004.08.001

Inoue, Y., Ikeda, M., & Shimizu, T. (2004). Proteome-wide classification and identification of mammalian-type GPCRs by binary topology pattern. *Computational Biology and Chemistry*, *28*(1), 39–49. https://doi.org/10.1016/j.compbiolchem.2003.11.003

Iqbal, M. J., Faye, I., & Samir, B. B. (2016). Classification of GPCRs proteins using a statistical encoding method. *Proceedings of the International Joint Conference on Neural Networks*, *2016–Octob*, 1224–1228. https://doi.org/10.1109/IJCNN.2016.7727337

Isberg, V., Mordalski, S., Munk, C., Rataj, K., Harpsøe, K., Hauser, A. S., … Gloriam, D. E. (2016). GPCRdb: an information system for G protein-coupled receptors. *Nucleic Acids Research*, *44*(D1), D356-64. https://doi.org/10.1093/nar/gkv1178

Jo, M., & Jung, S. T. (2016). Engineering therapeutic antibodies targeting G-protein–coupled receptors. *Experimental & Molecular Medicine*, *48*(2), e207. https://doi.org/10.1038/emm.2015.105

Karchin, R., Karplus, K., & Haussler, D. (2002). Classifying G-protein coupled receptors with support vector machines. *Bioinformatics (Oxford, England)*, *18*(1), 147–159. https://doi.org/10.1093/bioinformatics/18.1.147

Krishnan, A., Almén, M. S., Fredriksson, R., & Schiöth, H. B. (2012). The origin of GPCRs: identification of mammalian like Rhodopsin, Adhesion, Glutamate and Frizzled GPCRs in fungi. *PloS One*, *7*(1), e29817. https://doi.org/10.1371/journal.pone.0029817

Langenhan, T., Aust, G., & Hamann, J. (2013). Sticky signaling--adhesion class G protein-coupled receptors take the stage. *Science Signaling*, *6*(276), re3. https://doi.org/10.1126/scisignal.2003825

Li, Y. H., Xu, J. Y., Tao, L., Li, X. F., Li, S., Zeng, X., … Chen, Y. Z. (2016). SVM-Prot 2016: A Web-Server for Machine Learning Prediction of Protein Functional Families from Sequence Irrespective of Similarity. *PLOS ONE*, *11*(8), e0155290. https://doi.org/10.1371/journal.pone.0155290

Li, Z., Zhou, X., Dai, Z., & Zou, X. (2010). Classification of G-protein coupled receptors based on support vector machine with maximum relevance minimum redundancy and genetic algorithm. *BMC Bioinformatics*, *11*, 325. https://doi.org/10.1186/1471-2105-11-325

Liao, Z., Ju, Y., & Zou, Q. (2016). Prediction of G Protein-Coupled Receptors with SVM-Prot Features and Random Forest. *Scientifica*, *2016*, 1–10. https://doi.org/10.1155/2016/8309253

Liu, Y., An, S., Ward, R., Yang, Y., Guo, X.-X., Li, W., & Xu, T.-R. (2016). G protein-coupled receptors as promising cancer targets. *Cancer Letters*, *376*(2), 226–239. https://doi.org/10.1016/j.canlet.2016.03.031

Munoz, S., Guerrero, F. D., Kellogg, A., Heekin, A. M., & Leung, M. Y. (2017). Bioinformatic prediction of G protein-coupled receptor encoding sequences from the transcriptome of the foreleg, including the Haller's organ, of the cattle tick, Rhipicephalus australis. *PLoS ONE*, *12*(2), 1–22. https://doi.org/10.1371/journal.pone.0172326

Naveed, M., & Khan, A. U. (2012). GPCR-MPredictor: Multi-level prediction of G protein-coupled receptors using genetic ensemble. *Amino Acids*, *42*(5), 1809–1823. https://doi.org/10.1007/s00726-011-0902-6

Palczewski, K., Kumasaka, T., Hori, T., Behnke, C. A., Motoshima, H., Fox, B. A., … Miyano, M. (2000). Crystal Structure of Rhodopsin: A G Protein-Coupled Receptor. *Science*, *289*(5480), 739–745. https://doi.org/10.1126/science.289.5480.739

Pearson, W. R., & Lipman, D. J. (1988). Improved tools for biological sequence comparison. *Proceedings of the National Academy of Sciences of the United States of America*, *85*(8), 2444–8. Retrieved from http://www.ncbi.nlm.nih.gov/pubmed/3162770

Peng, Z.-L., Yang, J.-Y., & Chen, X. (2010). An improved classification of G-protein-coupled receptors using sequence-derived features. *BMC Bioinformatics*, *11*(1), 420. https://doi.org/10.1186/1471-2105-11-420

Poyner, D. R., & Hay, D. L. (2012). Secretin family (Class B) G protein-coupled receptors - from molecular to clinical perspectives. *British Journal of Pharmacology*, *166*(1), 1–3. https://doi.org/10.1111/j.1476-5381.2011.01810.x

Qian, B., Soyer, O. S., Neubig, R. R., & Goldstein, R. A. (2003). Depicting a protein's two faces: GPCR classification by phylogenetic tree-based HMMs. *FEBS Letters*, *554*(1–2), 95–99. https://doi.org/10.1016/S0014-5793(03)01112-8

Raisley, B., Zhang, M., Hereld, D., & Hadwiger, J. A. (2004). A cAMP receptor-like G protein-coupled receptor with roles in growth regulation and development. *Developmental Biology*, *265*(2), 433–45. Retrieved from http://www.ncbi.nlm.nih.gov/pubmed/14732403

Sahin, M. E., Can, T., & Son, C. D. (2014). GPCRsort-responding to the next generation

sequencing data challenge: prediction of G protein-coupled receptor classes using only structural region lengths. *Omics: A Journal of Integrative Biology*, *18*(10), 636–644. https://doi.org/10.1089/omi.2014.0073

Schiöth, H. B., & Fredriksson, R. (2005). The GRAFS classification system of G-protein coupled receptors in comparative perspective. *General and Comparative Endocrinology*, *142*(1–2 SPEC. ISS.), 94–101. https://doi.org/10.1016/j.ygcen.2004.12.018

Stockert, J. A., & Devi, L. A. (2015). Advancements in therapeutically targeting orphan GPCRs. *Frontiers in Pharmacology*, *6*, 100. https://doi.org/10.3389/fphar.2015.00100

UniProt Consortium, T. U. (2008). The universal protein resource (UniProt). *Nucleic Acids Research*, *36*(Database issue), D190-5. https://doi.org/10.1093/nar/gkm895

Venkatakrishnan, A. J., Deupi, X., Lebon, G., Heydenreich, F. M., Flock, T., Miljus, T., … Babu, M. M. (2016). Diverse activation pathways in class A GPCRs converge near the G-protein-coupling region. *Nature*, *536*(7617), 484–487. https://doi.org/10.1038/nature19107

Wu, H., Wang, C., Gregory, K. J., Han, G. W., Cho, H. P., Xia, Y., … Stevens, R. C. (2014). Structure of a class C GPCR metabotropic glutamate receptor 1 bound to an allosteric modulator. *Science (New York, N.Y.)*, *344*(6179), 58–64. https://doi.org/10.1126/science.1249489

Xiao, X., Wang, P., & Chou, K.-C. (2009). GPCR-CA: A cellular automaton image approach for predicting G-protein-coupled receptor functional classes. *Journal of Computational Chemistry*, *30*(9), 1414–1423. https://doi.org/10.1002/jcc.21163

Zamanian, M., Kimber, M. J., McVeigh, P., Carlson, S. A., Maule, A. G., & Day, T. A. (2011). The repertoire of G protein-coupled receptors in the human parasite Schistosoma mansoni and the model organism Schmidtea mediterranea. *BMC Genomics*, *12*, 596. https://doi.org/10.1186/1471-2164-12-596

Appendices

Appendix A – Percentages of amino acids in Python

```
import sys,subprocess , numpy
def count_aa():
      infile = open('non_five.fasta', 'r')
      lines = infile.readlines()
      seqs = []
      tempseq = ''
      for line in lines:
            if line[0] == '>' and not tempseq:
                  tempheader = line.strip()
            elif line[0] == '>':
                  seqs.append([tempheader,tempseq])
                  tempheader = line.strip()
                  tempseq = ''
            else:
                  tempseq = tempseq + line.strip()
      seqs.append([tempheader,tempseq])

# create a file with tm = 6 or 8
      tmhmm_file = open('tmhmm.fasta','w')
      fo = open('swiss_id.csv','a')
#For each sequence
      for seq in seqs:
      #Write to temp file
            tempfile = open('temp.fasta', 'w')
            tempfile.write('%s\n%s'%(seq[0],seq[1]))
            tempfile.close()

      #Write to temp file call tmhmm


            comm = '/export/home/kbegum/gpcr_code/tmhmm-2.0c/bin/tmhmm
temp.fasta -noplot'
            process = subprocess.Popen(comm, stdout=subprocess.PIPE,
stderr=subprocess.PIPE, shell=True)
            process.wait()
            lin = process.stdout.readlines()

            with open("temp.fasta") as gpcr:

                  counts = {}
                  keys = ["AA", "C", "D", "E", "F", "G", "H", "I", "K",
"L", "M", "N", "P", "Q", "R", "S", "T", "V", "W", "Y"]
                  keys1 = ["X"]
                  for char in keys:
                        counts[char] = 0

                  for line in gpcr:
                        if line.startswith(">"):
```

```
                    line = line.replace(' ','')
                    header = line.split()
                    number = header[0].split('|')

               else:
                    seq_length = len(line)
                    z = str(seq_length)
                    print seq_length
               for char in line.strip():


                    if char in keys:
                         counts[char] += 1
                    total = float(sum(counts.values()))
                    others = seq_length - total
                    s_seq_length = str(others)


          toReturn = ''


          for key in keys:
               aa_per = (counts[key])
               toReturn = toReturn + '%.4f'%aa_per + ','

          fo.write(''.join(str(x) for x in toReturn))
          #fo.write(s_seq_length)
          fo.write('\n')
          return toReturn
     fo.close()

count_aa()
```

Appendix B – Non-GPCR IDs, Protein name in Python

```
import sys,subprocess , numpy
def nongpcr_id():
      infile = open('Non.fasta', 'r')
      lines = infile.readlines()
      seqs = []
      tempseq = ''
      for line in lines:
            if line[0] == '>' and not tempseq:
                  tempheader = line.strip()
            elif line[0] == '>':
                  seqs.append([tempheader,tempseq])
                  tempheader = line.strip()
                  tempseq = ''
            else:
                  tempseq = tempseq + line.strip()
      seqs.append([tempheader,tempseq])

# create a file for the IDs
      fo = open('non_cdhit_ID.csv','a')
#For each sequence
      for seq in seqs:
      #Write to temp file
            tempfile = open('temp.fasta', 'w')
            tempfile.write('%s\n%s'%(seq[0],seq[1]))
            tempfile.close()

            with open("temp.fasta") as gpcr:

                  for line in gpcr:
                        if line.startswith(">"):
                              line = line.replace(' ','')
                              header = line.split()
                              number = header[0].split('|')

                              print "Id:",number[1]

                              continue

                  fo.write(number[1])
                  fo.write('\n')

      fo.close()

nongpcr_id()


import sys,subprocess , numpy
def protein_name():
      infile = open('Non.fasta', 'r')
      lines = infile.readlines()
```

```
      seqs = []
      tempseq = ''
      for line in lines:
            if line[0] == '>' and not tempseq:
                  tempheader = line.strip()
            elif line[0] == '>':
                  seqs.append([tempheader,tempseq])
                  tempheader = line.strip()
                  tempseq = ''
            else:
                  tempseq = tempseq + line.strip()
      seqs.append([tempheader,tempseq])

# create a file for protein names
      fo = open('non_cdhit_protein_name.csv','a')
#For each sequence
      for seq in seqs:
      #Write to temp file
            tempfile = open('temp.fasta', 'w')
            tempfile.write('%s\n%s'%(seq[0],seq[1]))
            tempfile.close()


            with open("temp.fasta") as gpcr:

                  for line in gpcr:
                        if line.startswith(">"):
                              header = line.split('|')
                              number = header[2].split(' ')

                              print number[1]

                              continue

                  fo.write(number[1])
                  fo.write('\n')

      fo.close()

protein_name()

import sys,subprocess , numpy
def sequence():
      infile = open('Non.fasta', 'r')
      lines = infile.readlines()
      seqs = []
      tempseq = ''
      for line in lines:
            if line[0] == '>' and not tempseq:
                  tempheader = line.strip()
            elif line[0] == '>':
                  seqs.append([tempheader,tempseq])
```

```
                tempheader = line.strip()
                tempseq = ''
            else:
                tempseq = tempseq + line.strip()
        seqs.append([tempheader,tempseq])

# Create a file for the sequences
        fo = open('non_cdhit_sequences.csv','a')
#For each sequence
        for seq in seqs:
        #Write to temp file
            tempfile = open('temp.fasta', 'w')
            tempfile.write('%s\n%s'%(seq[0],seq[1]))
            tempfile.close()


            with open("temp.fasta") as gpcr:

                for line in gpcr:
                    if line.startswith(">"):

                        line = line.replace(' ','')
                        header = line.split()
                        number = header[0].split('|')


                        continue

                    else:
                        seq_length = len(line)
                        fo.write(line)
                        fo.write('\n')
                        print line


        fo.close()

sequence()
```

Appendix C – Dipeptide count of amino acids in Python

Dipeptide count code:

```
import re
complete=[]
count=0
seq=""
for line in open("Non.fasta"):
    if line.startswith(">"):
        if count>0:
            complete.append(seq)
        header = line
        header = header.replace("\n","")
        header = header.replace("\r","")
        complete.append(header)
        seq=""
        count=count+1
    else:
        hold = line
        hold = hold.replace("\n","")
        hold = hold.replace("\r","")
        seq = seq + hold

complete.append(seq)

nucleotides = ["A", "C", "D", "E", "F", "G", "H", "I", "K", "L", "M",
"N", "P", "Q", "R", "S", "T", "V", "W", "Y"]
'''
words1 = []
for x in nucleotides:
        word=x
        words1.append(word)

for e in complete:
    for n in words1:
        if e.startswith(">"):
            header = e
        else:
            num=e.count(n)
            file = open("1_words.csv","a")
            header = header.replace(",",";")
            l=len(e)
            freq = (num/float(l))
            file.write(header + "," + str(l) + "," + n + "," +
str(num) + "," + str(freq) + "\n")
            file.close()
'''
file = open("2_mist1.csb","a")
words2 = []
p = []
```

```
for x in nucleotides:
    for y in nucleotides:
        word=x+y
#     word = word + ','
        words2.append(word)

#print str.replace("'","")

for e in complete:
    length = len(e)
    for n in words2:
        if e.startswith(">"):
            header = e
        else:
            num = 0
            idx = 0
            while True:
                idx = e.find(n, idx)
                if idx >= 0:
                    num += 1
                    idx += 1
                else:
                    break
            p.append(num)

     chunks = [p[x:x+400] for x in range(0, len(p), 400)]

c = 0
for i in chunks:
     file.write(str(chunks[c]))
     file.write('\n')

     c = c + 1
file.close()
```

Appendix D – Taxonomy, IDs for Organisms in Python

```
import sys,subprocess , numpy

def strain():

      infile = open('SwissProtGPCRs.fasta', 'r')
      lines = infile.readlines()
      seqs = []
      tempseq = ''
      for line in lines:
            if line[0] == '>' and not tempseq:
                  tempheader = line.strip()
            elif line[0] == '>':
                  seqs.append([tempheader,tempseq])
                  tempheader = line.strip()
                  tempseq = ''
            else:
                  tempseq = tempseq + line.strip()
      seqs.append([tempheader,tempseq])


# create a file for strain
      fo1 = open('11_13_gpcr_organism_name.csv','a')
      fo = open('11_13_gpcr_virus_name.csv','a')
#For each sequence
      for seq in seqs:
      #Write to temp file
            tempfile = open('temp.fasta', 'w')
            tempfile.write('%s\n%s'%(seq[0],seq[1]))
            tempfile.close()


            with open("temp.fasta") as gpcr:

                  xlist = []
                  for line in gpcr:
                        if line.startswith(">"):
                        #     print line
                              header = line.split('=')
                              #organism = header[1].split('GN')[0]
                              organism = header[1][0:-3]
                              column1 = header[0].split('|')
                              column1 = column1[1]
                              altprint = 0
                              sero = ''
                              strain = ''
                              if 'virus' in organism or 'Virus' in
organism:
                                    altprint = 1
                              elif '(' in organism:
                                    ols = organism.split('(')
```

```
                                    gss = ols[0].strip().split()
                                    if len(gss) == 2:
                                          genus = gss[0]
                                          species = gss[1]
                                    elif len(gss) > 2:
                                          genus = gss[0]
                                          species = gss[1]
                                          sero = ' '.join(gss[2:])
                                    else:
                                          altprint = 1

                                    strain = ols[1].split(')')[0]
                              else:
                                    gss = organism.strip().split()
                                    if len(gss) == 2:
                                          genus = gss[0]
                                          species = gss[1]
                                    elif len(gss) > 2:
                                          genus = gss[0]
                                          species = gss[1]
                                          sero = ' '.join(gss[2:])
                                    else:
                                           altprint = 1
                              if altprint == 1:
                                    fo.write(','.join([column1,organism]))
                                    fo.write('\n')
                              else:

      fo1.write(','.join([column1,genus,species,sero,strain]))
                                    fo1.write('\n')

      fo.close()
      fo1.close()
strain()
```

Organism table ID adjustments:

```
import re
infile1 = open('11_29_id_organism.csv','r')
infile2 = open('11_29_organism_table.csv','r')
lines1 = infile1.read().splitlines()
lines2 = infile2.readlines()

fo = open('1130_organism.csv','a')
d = ''

#m = [m.replace('\r\n','') for m in lines1]


for line in lines1:
      m = line.split(',')
      for lines in lines2:
            c = lines.split(',')
```

```
            if m[1:5] == c[1:5]:
                  d = str(m[0]) + ',' + str(c[0])
                  print d


                  fo.write(d)
                  fo.write('\n')
fo.close()
```

Appendix E – Length using TMHMM2.0 in Python

```
import sys,subprocess , numpy
def length_aa():
      infile = open('non_five.fasta', 'r')
      lines = infile.readlines()
      seqs = []
      tempseq = ''
      for line in lines:
            if line[0] == '>' and not tempseq:
                  tempheader = line.strip()
            elif line[0] == '>':
                  seqs.append([tempheader,tempseq])
                  tempheader = line.strip()
                  tempseq = ''
            else:
                  tempseq = tempseq + line.strip()
      seqs.append([tempheader,tempseq])

# create a file with tm = 6 or 8
      tmhmm_file = open('tmhmm.fasta','w')
      #fo = open('aa_count_mist_swiss.csv','a')
#For each sequence
      for seq in seqs:
      #Write to temp file
            tempfile = open('temp.fasta', 'w')
            tempfile.write('%s\n%s'%(seq[0],seq[1]))
            tempfile.close()

      #Write to temp file call tmhmm


            comm = '/export/home/kbegum/gpcr_code/tmhmm-2.0c/bin/tmhmm
temp.fasta -noplot'
            process = subprocess.Popen(comm, stdout=subprocess.PIPE,
stderr=subprocess.PIPE, shell=True)
            process.wait()
            lin = process.stdout.readlines()

            with open("temp.fasta") as gpcr:

                  counts = {}
                  keys = ["A", "C", "D", "E", "F", "G", "H", "I", "K",
"L", "M", "N", "P", "Q", "R", "S", "T", "V", "W", "Y"]
                  keys1 = ["X"]
                  for char in keys:
                        counts[char] = 0

                  for line in gpcr:
                        if line.startswith(">"):
                              line = line.replace(' ','')
                              header = line.split()
```

```
                        number = header[0].split('|')

                #       print number[2]

                        continue
                #       fo.write(number[1])
                #       fo.write('\n')
                    else:
                        seq_length = len(line)
                        z = str(seq_length)
            #           fo.write(z)
            #           fo.write('\n')
                        #print seq_length
                    for char in line.strip():


                        if char in keys:
                            counts[char] += 1
                        total = float(sum(counts.values()))
                        others = seq_length - total
                        s_seq_length = str(others)


                toReturn = ''


                for key in keys:
                    aa_per = (counts[key])
                    toReturn = toReturn + str(aa_per) + ','
                #fo.write('ID')
                #fo.write('\t')
                #fo.write(number[2])

                #print toReturn
                #fo.write('\n')
    #           fo.write(''.join(str(x) for x in toReturn))
                #fo.write(s_seq_length)
    #           fo.write('\n')

                print toReturn
#               return toReturn
    fo.close()

length_aa()
```

Appendix F – MySQL commands for creating tables and populating data

PROTEINS table:

```
LOAD DATA LOCAL INFILE
'/export/home/kbegum/gpcr_code/merge_gpcr_final_1.csv' INTO TABLE
GPCR2 FIELDS TERMINATED BY ',' LINES TERMINATED BY '\n' (PROTEIN_ID,
Name,Organism_ID,IUPHAR_ID,GRAFS_ID,Sequence,Length,A,C,D,E,F,G,H,I,K,
L,M,N,P,Q,R,S,T,V,W,Y,Other,PROTEIN_TYPE);
```

Dipeptide table:

```
LOAD DATA LOCAL INFILE
'/export/home/kbegum/gpcr_code/merge_gpcr_final_1.csv' INTO TABLE
GPCR2 FIELDS TERMINATED BY ',' LINES TERMINATED BY '\n' (GPCR_ID,
Name,Organism_ID,IUPHAR_ID,GRAFS_ID,Sequence,Length,A,C,D,E,F,G,H,I,K,
L,M,N,P,Q,R,S,T,V,W,Y,Other,PROTEIN_TYPE);
CREATE TABLE AA_Dipeptide( PROTEIN_ID varchar(25) PRIMARY KEY, A1
varchar(25), C1 varchar(25), D1 varchar(25), E1 varchar(25), F1
varchar(25), G1 varchar(25), H1 varchar(25), I1 varchar(25), K1
varchar(25), L1 varchar(25), M1 varchar(25), N1 varchar(25), P1
varchar(25), Q1 varchar(25), R1 varchar(25), S1 varchar(25), T1
varchar(25), V1 varchar(25), W1 varchar(25), Y1 varchar(25), Others1
varchar(25), AA varchar(25),  AC varchar(25),  AD varchar(25),  AE
varchar(25),  AF varchar(25),  AG varchar(25),  AH varchar(25),  AI
varchar(25),  AK varchar(25),AL varchar(25),AM varchar(25), AN
varchar(25),  AP varchar(25),  AQ varchar(25),  AR varchar(25),  AS1
varchar(25),  AT varchar(25),  AV varchar(25),  AW varchar(25),  AY
varchar(25),  CA varchar(25),  CC varchar(25),  CD varchar(25),  CE
varchar(25),  CF varchar(25),  CG varchar(25),  CH varchar(25),  CI
varchar(25),  CK varchar(25),  CL varchar(25),  CM varchar(25),  CN
varchar(25),  CP varchar(25),  CQ varchar(25),  CR varchar(25),  CS
varchar(25),  CT varchar(25),  CV varchar(25),  CW varchar(25),  CY
varchar(25),  DA varchar(25),  DC varchar(25),  DD varchar(25),  DE
varchar(25),  DF varchar(25),  DG varchar(25),  DH varchar(25),  DI
varchar(25),  DK varchar(25),  DL varchar(25),  DM varchar(25),  DN
varchar(25),  DP varchar(25),  DQ varchar(25),  DR varchar(25),  DS
varchar(25),  DT varchar(25),  DV varchar(25),  DW varchar(25),  DY
varchar(25),  EA varchar(25),  EC varchar(25),  ED varchar(25),  EE
varchar(25),  EF varchar(25),  EG varchar(25),  EH varchar(25),  EI
varchar(25),  EK varchar(25),  EL varchar(25),  EM varchar(25),  EN
varchar(25),  EP varchar(25),  EQ varchar(25),  ER varchar(25),  ES
varchar(25),  ET varchar(25),  EV varchar(25),  EW varchar(25),  EY
varchar(25),  FA varchar(25),  FC varchar(25),  FD varchar(25),  FE
varchar(25),  FF varchar(25),  FG varchar(25),  FH varchar(25),  FI
varchar(25),  FK varchar(25),  FL varchar(25),  FM varchar(25),  FN
varchar(25),  FP varchar(25),  FQ varchar(25),  FR varchar(25),  FS
varchar(25),  FT varchar(25),  FV varchar(25),  FW varchar(25),  FY
varchar(25),  GA varchar(25),  GC varchar(25),  GD varchar(25),  GE
varchar(25),  GF varchar(25),  GG varchar(25),  GH varchar(25),  GI
varchar(25),  GK varchar(25),  GL varchar(25),  GM varchar(25),  GN
varchar(25),  GP varchar(25),  GQ varchar(25),  GR varchar(25),  GS
varchar(25),  GT varchar(25),  GV varchar(25),  GW varchar(25),  GY
varchar(25),  HA varchar(25),  HC varchar(25),  HD varchar(25),  HE
```

```
varchar(25),  HF varchar(25),  HG varchar(25),  HH varchar(25),  HI
varchar(25),  HK varchar(25),  HL varchar(25),  HM varchar(25),  HN
varchar(25),  HP varchar(25),  HQ varchar(25),  HR varchar(25),  HS
varchar(25),  HT varchar(25),  HV varchar(25),  HW varchar(25),  HY
varchar(25),  IA varchar(25),  IC varchar(25),  ID varchar(25),  IE
varchar(25),  IF1 varchar(25),  IG varchar(25),  IH varchar(25),  II
varchar(25),  IK varchar(25),  IL varchar(25),  IM varchar(25),  IN1
varchar(25),  IP varchar(25),  IQ varchar(25),  IR varchar(25),  IS1
varchar(25),  IT varchar(25),  IV varchar(25),  IW varchar(25),  IY
varchar(25),  KA varchar(25),  KC varchar(25),  KD varchar(25),  KE
varchar(25),  KF varchar(25),  KG varchar(25),  KH varchar(25),  KI
varchar(25),  KK varchar(25),  KL varchar(25),  KM varchar(25),  KN
varchar(25),  KP varchar(25),  KQ varchar(25),  KR varchar(25),  KS
varchar(25),  KT varchar(25),  KV varchar(25),  KW varchar(25),  KY
varchar(25),  LA varchar(25),  LC varchar(25),  LD varchar(25),  LE
varchar(25),  LF varchar(25),  LG varchar(25),  LH varchar(25),  LI
varchar(25),  LK varchar(25),  LL varchar(25),  LM varchar(25),  LN
varchar(25),  LP varchar(25),  LQ varchar(25),  LR varchar(25),  LS
varchar(25),  LT varchar(25),  LV varchar(25),  LW varchar(25),  LY
varchar(25),  MA varchar(25),  MC varchar(25),  MD varchar(25),  ME
varchar(25),  MF varchar(25),  MG varchar(25),  MH varchar(25),  MI
varchar(25),  MK varchar(25),  ML varchar(25),  MM varchar(25),  MN
varchar(25),  MP varchar(25),  MQ varchar(25),  MR varchar(25),  MS
varchar(25),  MT varchar(25),  MV varchar(25),  MW varchar(25),  MY
varchar(25),  NA varchar(25),  NC varchar(25),  ND varchar(25),  NE
varchar(25),  NF varchar(25),  NG varchar(25),  NH varchar(25),  NI
varchar(25),  NK varchar(25),  NL varchar(25),  NM varchar(25), NN
varchar(25),  NP varchar(25),  NQ varchar(25),  NR varchar(25),  NS
varchar(25),  NT varchar(25),  NV varchar(25),  NW varchar(25),  NY
varchar(25),  PA varchar(25),  PC varchar(25),  PD varchar(25),  PE
varchar(25),  PF varchar(25),  PG varchar(25),  PH varchar(25),  PI
varchar(25),  PK varchar(25),  PL varchar(25),  PM varchar(25),  PN
varchar(25),  PP varchar(25),  PQ varchar(25),  PR varchar(25),  PS
varchar(25),  PT varchar(25),  PV varchar(25),  PW varchar(25),  PY
varchar(25),  QA varchar(25),  QC varchar(25),  QD varchar(25),  QE
varchar(25),  QF varchar(25),  QG varchar(25),  QH varchar(25),  QI
varchar(25),  QK varchar(25), QL varchar(25),  QM varchar(25),  QN
varchar(25),  QP varchar(25),  QQ varchar(25),  QR varchar(25),  QS
varchar(25),  QT varchar(25),  QV varchar(25),  QW varchar(25),  QY
varchar(25),  RA varchar(25),  RC varchar(25),  RD varchar(25),  RE
varchar(25),  RF varchar(25),  RG varchar(25),  RH varchar(25),  RI
varchar(25),  RK varchar(25),  RL varchar(25),  RM varchar(25),  RN
varchar(25),  RP varchar(25),  RQ varchar(25),  RR varchar(25),  RS
varchar(25),  RT varchar(25),  RV varchar(25),  RW varchar(25),  RY
varchar(25),  SA varchar(25),  SC varchar(25),  SD varchar(25),  SE
varchar(25),  SF varchar(25),  SG varchar(25),  SH varchar(25),  SI
varchar(25),  SK varchar(25),  SL varchar(25),  SM varchar(25),  SN
varchar(25),  SP varchar(25),  SQ varchar(25),  SR varchar(25),  SS
varchar(25),  ST varchar(25),  SV varchar(25),  SW varchar(25),  SY
varchar(25),  TA varchar(25),  TC varchar(25),  TD varchar(25),  TE
varchar(25),  TF varchar(25),  TG varchar(25),  TH varchar(25),  TI
varchar(25),  TK varchar(25),  TL varchar(25),  TM varchar(25),  TN
```

```
varchar(25),  TP varchar(25),  TQ varchar(25),  TR varchar(25),  TS
varchar(25),  TT varchar(25),  TV varchar(25),  TW varchar(25),  TY
varchar(25),  VA varchar(25),  VC varchar(25),  VD varchar(25),  VE
varchar(25),  VF varchar(25),  VG varchar(25),  VH varchar(25),  VI
varchar(25),  VK varchar(25),  VL varchar(25),  VM varchar(25),  VN
varchar(25), VP varchar(25),  VQ varchar(25),  VR varchar(25),  VS
varchar(25),  VT varchar(25),  VV varchar(25),  VW varchar(25),  VY
varchar(25),  WA varchar(25),  WC varchar(25),  WD varchar(25),  WE
varchar(25),  WF varchar(25),  WG varchar(25),  WH varchar(25),  WI
varchar(25),  WK varchar(25),  WL varchar(25),  WM varchar(25),  WN
varchar(25),  WP varchar(25),  WQ varchar(25),  WR varchar(25),  WS
varchar(25),  WT varchar(25),  WV varchar(25),  WW varchar(25),  WY
varchar(25),  YA varchar(25),  YC varchar(25),  YD varchar(25),  YE
varchar(25),  YF varchar(25),  YG varchar(25),  YH varchar(25),  YI
varchar(25),  YK varchar(25),  YL varchar(25),  YM varchar(25),  YN
varchar(25),  YP varchar(25),  YQ varchar(25),  YR varchar(25),  YS
varchar(25),  YT varchar(25),  YV varchar(25),  YW varchar(25),  YY
varchar(25));


LOAD DATA LOCAL  INFILE '/export/home/kbegum/gpcr_code/dekhajak1.csv'
INTO TABLE AA_Dipeptide FIELDS TERMINATED BY ',' LINES TERMINATED BY
'\n'
(PROTEIN_ID,A1,C1,D1,E1,F1,G1,H1,I1,K1,L1,M1,N1,P1,Q1,R1,S1,T1,V1,W1,Y
1,Others1,AA, AC, AD, AE, AF, AG, AH, AI, AK, AL, AM, AN, AP, AQ, AR,
AS1, AT, AV, AW, AY, CA, CC, CD, CE, CF, CG, CH, CI, CK, CL, CM, CN,
CP, CQ, CR, CS, CT, CV, CW, CY, DA, DC, DD, DE, DF, DG, DH, DI, DK,
DL, DM, DN, DP, DQ, DR, DS, DT, DV, DW, DY, EA, EC, ED, EE, EF, EG,
EH, EI, EK, EL, EM, EN, EP, EQ, ER, ES, ET, EV, EW, EY, FA, FC, FD,
FE, FF, FG, FH, FI, FK, FL, FM, FN, FP, FQ, FR, FS, FT, FV, FW, FY,
GA, GC, GD, GE, GF, GG, GH, GI, GK, GL, GM, GN, GP, GQ, GR, GS, GT,
GV, GW, GY, HA, HC, HD, HE, HF, HG, HH, HI, HK, HL, HM, HN, HP, HQ,
HR, HS, HT, HV, HW, HY, IA, IC, ID, IE, IF1, IG, IH, II, IK, IL, IM,
IN1, IP, IQ, IR, IS1, IT, IV, IW, IY, KA, KC, KD, KE, KF, KG, KH, KI,
KK, KL, KM, KN, KP, KQ, KR, KS, KT, KV, KW, KY, LA, LC, LD, LE, LF,
LG, LH, LI, LK, LL, LM, LN, LP, LQ, LR, LS, LT, LV, LW, LY, MA, MC,
MD, ME, MF, MG, MH, MI, MK, ML, MM, MN, MP, MQ, MR, MS, MT, MV, MW,
MY, NA, NC, ND, NE, NF, NG, NH, NI, NK, NL, NM, NN, NP, NQ, NR, NS,
NT, NV, NW, NY, PA, PC, PD, PE, PF, PG, PH, PI, PK, PL, PM, PN, PP,
PQ, PR, PS, PT, PV, PW, PY, QA, QC, QD, QE, QF, QG, QH, QI, QK, QL,
QM, QN, QP, QQ, QR, QS, QT, QV, QW, QY, RA, RC, RD, RE, RF, RG, RH,
RI, RK, RL, RM, RN, RP, RQ, RR, RS, RT, RV, RW, RY, SA, SC, SD, SE,
SF, SG, SH, SI, SK, SL, SM, SN, SP, SQ, SR, SS, ST, SV, SW, SY, TA,
TC, TD, TE, TF, TG, TH, TI, TK, TL, TM, TN, TP, TQ, TR, TS, TT, TV,
TW, TY, VA, VC, VD, VE, VF, VG, VH, VI, VK, VL, VM, VN, VP, VQ, VR,
VS, VT, VV, VW, VY, WA, WC, WD, WE, WF, WG, WH, WI, WK, WL, WM, WN,
WP, WQ, WR, WS, WT, WV, WW, WY, YA, YC, YD, YE, YF, YG, YH, YI, YK,
YL, YM, YN, YP, YQ, YR, YS, YT, YV, YW, YY);

CREATE TABLE Organism_v2 (Organism_ID varchar(25) PRIMARY KEY, Genus
varchar(25) NULL, Species varchar(60) NULL, Serotype varchar(60) NULL,
```

```
Strain varchar(100) NULL, Common_name varchar(100) NULL, Frequency
varchar(25));
LOAD DATA LOCAL  INFILE
'/export/home/kbegum/gpcr_code/11_18_new_organismid.csv' INTO TABLE
Organism_v2 FIELDS TERMINATED BY ',' LINES TERMINATED BY '\r\n'
(Organism_ID, Genus, Species, Serotype, Strain, Common_name,
Frequency);
```

Updating Organism table:

```
CREATE TABLE Organism_v2 (Organism_ID varchar(25) PRIMARY KEY, Genus
varchar(25) NULL, Species varchar(60) NULL, Serotype varchar(60) NULL,
Strain varchar(100) NULL, Common_name varchar(100) NULL, Frequency
varchar(25));
LOAD DATA LOCAL  INFILE
'/export/home/kbegum/gpcr_code/11_18_new_organismid.csv' INTO TABLE
Organism_v2 FIELDS TERMINATED BY ',' LINES TERMINATED BY '\r\n'
(Organism_ID, Genus, Species, Serotype, Strain, Common_name,
Frequency);
```

Updating columns of Organism ID:

```
UPDATE PROTEINS SET Organism_ID = lpad(Organism_ID,5,0)
CREATE TABLE temp_table (PROTEIN_ID varchar(25) PRIMARY KEY,
Organism_ID varchar(25) NULL, Frequency varchar(25));
LOAD DATA LOCAL  INFILE
'/export/home/kbegum/gpcr_code/1130_organism_sql.csv' INTO TABLE
temp_table FIELDS TERMINATED BY ',' LINES TERMINATED BY '\r\n'
(PROTEIN_ID,Organism_ID);

UPDATE PROTEINS
INNER JOIN temp_table on temp_table.PROTEIN_ID = PROTEINS.PROTEIN_ID
SET PROTEINS.Organism_ID = temp_table.Organism_ID;

DROP TEMPORARY TABLE your_temp_table;
```

TMHMM length table:

```
CREATE TABLE TMHMM_Length (PROTEIN_ID varchar(25) PRIMARY KEY, N_term
varchar(25),IL1 varchar(25),IL2 varchar(25),IL3 varchar(25),TM1
varchar(25),TM2 varchar(25),TM3 varchar(25),TM4 varchar(25),TM5
varchar(25),TM6 varchar(25),TM7 varchar(25),OL1 varchar(25),OL2
varchar(25),OL3 varchar(25),C_term varchar(25),Length varchar(25));
LOAD DATA LOCAL  INFILE
'/export/home/kbegum/gpcr_code/11_28_tmhmm_length.csv' INTO TABLE
TMHMM_Length FIELDS TERMINATED BY ',' LINES TERMINATED BY '\r\n'
(PROTEIN_ID, N_term, IL1, IL2,
IL3,TM1,TM2,TM3,TM4,TM5,TM6,TM7,OL1,OL2,OL3,C_term,Length);
```

Appendix G – R code for generating plots

R code for plots:

```
options(max.print=10000000)
data = read.csv("11_28_length_Rplot_file.csv", header = TRUE)
par(mfrow=c(2,2))
# N-terminus histogram
hist(data[,1], xlim= c(0, 1000),ylim= c(0,2000), xlab = "Length", ylab
= "Frequency", main = "N-terminus", breaks=100)

#Inside loop lengths histogram
hist(data[,2], xlim= c(0, 600),ylim= c(0,2500), xlab = "Length", ylab
= "Frequency", main = "Inside loop 1", breaks=10)
hist(data[,3], xlim = c(0,150), ylim= c(0,2500),xlab = "Length", ylab
= "Frequency", main = "Inside loop 2", breaks=10)
hist(data[,4], xlim = c(0,1000), ylim= c(0,2500),xlab = "Length", ylab
= "Frequency", main = "Inside loop 3",breaks = 10)

#TM
par(mfrow=c(2,4))
hist(data[,5], xlim= c(15, 35),ylim= c(0,2500), xlab = "Length", ylab
= "Frequency", main = "TMhelix 1",breaks=5)
hist(data[,6], xlim = c(15,35), ylim= c(0,2500),xlab = "Length", ylab
= "Frequency", main = "TMhelix 2", breaks=5)
hist(data[,7], xlim = c(15,30), ylim= c(0,2500),xlab = "Length", ylab
= "Frequency", main = "TMhelix 3",breaks = 5)
hist(data[,8], xlim= c(15, 30),ylim= c(0,2500), xlab = "Length", ylab
= "Frequency", main = "TMhelix 4", breaks=5)
hist(data[,9], xlim = c(10,40), ylim= c(0,2500),xlab = "Length", ylab
= "Frequency", main = "TMhelix 5", breaks=5)
hist(data[,10], xlim = c(15,30), ylim= c(0,2500),xlab = "Length", ylab
= "Frequency", main = "TMhelix 6",breaks = 5)
hist(data[,11], xlim = c(15,30), ylim= c(0,2500),xlab = "Length", ylab
= "Frequency", main = "TMhelix 7",breaks = 5)

par(mfrow=c(2,2))
#Outside loop lengths histogram
hist(data[,12], xlim= c(0, 80),ylim= c(0,1500), xlab = "Length", ylab
= "Frequency", main = "Outside loop 1", breaks=10)
hist(data[,13], xlim = c(0,200), ylim= c(0,1500),xlab = "Length", ylab
= "Frequency", main = "Outside loop 2", breaks=10)
hist(data[,14], xlim = c(0,60), ylim= c(0,1000),xlab = "Length", ylab
= "Frequency", main = "Outside loop 3",breaks = 10)


hist(data[,15], xlim = c(0,800), ylim= c(0,1500),xlab = "Length", ylab
= "Frequency", main = "C-terminus",breaks = 25)

# Sum of the length regions

options(max.print=10000000)
```

```
data = read.csv("swiss_fivepoint_length.csv", header = TRUE)
test = t(swiss_fivepoint_length)
test
boxplot(test,ylim=c(0,200),outline = FALSE, main = 'Boxplot of N-term,
Helix, Outside loop, Inside loop and C-term')
par(mfrow=c(2,3))

hist(test[,1], xlim= c(0, 1000),ylim= c(0,2000), xlab = "N-term", ylab
= "Frequency", main = "N-terminal histogram", breaks=100)
hist(test[,2], xlim= c(120, 180),ylim= c(0,1000), xlab = "Helix", ylab
= "Frequency", main = "Helix histogram", breaks=5)
hist(test[,3], xlim = c(0,250), ylim= c(0,1500),xlab = "Outside loop",
ylab = "Frequency", main = "outside loop histogram", breaks=10)
hist(test[,4], xlim = c(0,1000), ylim= c(0,2000),xlab = "Inside loop",
ylab = "Frequency", main = "inside loop histogram",breaks = 10)
hist(test[,5], xlim = c(0,800), ylim= c(0,2000),xlab = "C-term", ylab
= "Frequency", main = "C-terminal histogram",breaks = 25)

fivenum(test[,1])
fivenum(test[,2])
fivenum(test[,3])
fivenum(test[,4])
fivenum(test[,5])
```

Curriculum Vita

Khodeza Begum was born in Dhaka, Bangladesh. The youngest daughter of Md. Khorshed Alam and Monowara Begum, she got her secondary and higher secondary school certificates in 2005 and 2007 respectively from Viqarunnisa noon school and college in Dhaka, Bangladesh. She entered The American International University of Bangladesh in fall 2008 to pursue her bachelor's degree in Electrical and Electronics Engineering. She received 'Summa Cum Laude' for academic excellence and graduated in 2012. After pursuing her bachelor's degree, she started working as a lab instructor in the engineering department of North South University in Dhaka, Bangladesh for over a year where she taught programming languages and computer networking lab. In 2014, she worked as a lab officer in Management Information Sytem of the Business school in North South University and later she joined the Information technology department as an IT officer. Khodeza got accepted in the Computational Science program at University of Texas at El Paso to pursue her Doctoral degree in fall 2015 and currently resides in El Paso with her husband and one child.

Email address: kbegum@miners.utep.edu

This thesis was typed by Khodeza Begum.