Classifying influenza Subtypes and Hosts using Machine Learning Techniques

A Thesis

Presented to the

Department of Computer Science

University of Nebraska

In Partial Fulfillment
Of the Requirements for the Degree of

Master of Computer Science

University of Nebraska at Omaha

by

Pavan Kumar Attaluri

May 2010

Supervisory Committee:

Zhengxin Chen

Guoqing Lu

Parvathi Chundi

UMI Number: 1474929

All rights reserved

INFORMATION TO ALL USERS
The quality of this reproduction is dependent upon the quality of the copy submitted.

In the unlikely event that the author did not send a complete manuscript
and there are missing pages, these will be noted. Also, if material had to be removed,
a note will indicate the deletion.

UMI 1474929
Copyright 2010 by ProQuest LLC.
All rights reserved. This edition of the work is protected against
unauthorized copying under Title 17, United States Code.

ProQuest LLC
789 East Eisenhower Parkway
P.O. Box 1346
Ann Arbor, MI 48106-1346

Classifying influenza Subtypes and Hosts using Machine Learning Techniques

Pavan Kumar Attaluri, MS

University of Nebraska, 2010

Advisor: Zhengxin Chen

Recent advances in machine learning techniques have made its way into a wide variety of fields in an impressive way. There has been a tremendous amount of research going on the improvement of various machine learning methods. However, the study of utilizing machine learning techniques in a systematic way is meager. In this research, we explored a methodology for integrated use of various machine learning techniques for influenza analysis.

Influenza is one of the most important emerging and reemerging infectious diseases, causing high morbidity and mortality in communities and worldwide. Classification and prediction analysis helps to better understand the evolution of influenza virus and developing tools for detection of new viral strains. The main objective research is to classify the influenza A virus sequences based on host of origin and subtype, for which decision tree analysis, support vector machine (SVM) and artificial neural networks have been applied. A Web based tool is developed using hidden Markov model (HMM) for accurate prediction of origin and subtype.

With decision tree analysis, the accuracies of classification results varied between 93-97%. Informative positions are extracted from decision trees and modeled into profiles through hidden Markov modeling. These profiles are used in the Web prediction system. The host and subtype prediction system achieved 88% accuracy. With support vector machine analysis, the accuracies of classification results varied between 96-98%. With

neural networks, the accuracies of classification results varied between 88-94%. Mutation positions are found through studying the informative positions determined by the decision tree method at protein level and stored in a database.

This project paves the way for further experiments to examine the informative positions at protein level, extend its current functionality to classify more subtypes and host origins and investigate other advanced machine learning algorithms. Developing a Web tool for the prediction of all influenza A hosts and subtypes has significances in the development of a computational system for influenza detection and surveillance.

Acknowledgements

I would like to thank Dr. Guoqing Lu for sparking my interest in bioinformatics and for his guidance and encouragement throughout the thesis. My special thanks to my supervisors Dr. Zhengxin Chen and Dr. Parvathi Chundi, for their valuable suggestions and feedback. Special thanks to Ximeng Zheng for his help regarding support vector machine and Aruna Weerakoon for his help during initial stages of the project. I would also like to thank my colleagues in the Bioinformatics lab for their valuable discussions and suggestions. This research was supported by NIH grant numbers P20 RR016469 from the INBRE program of the National Center for Research Resources and R01 LM009985-01A1.

Finally, I would like to thank my family and friends for their continuous support.

Contents

List of Tables

List of Figures

Chapter 1

Introduction

The advance of computational techniques for solving biology problems is growing in importance. These techniques provide researchers an insight to solve analytical problems, such as genetic analysis and prediction. With the assistance of the automated learning and predictive modeling methods, researchers can analyze biological data with greater accuracy, in less time and at lower costs.

The enormous growth of biological data generated by high-throughput methods in biology requires new methods to analyze these data. The Human Genome Project completed in 2003, establishes high standards of algorithms and computational tools in understanding the human genome, fast sequence comparison, and efficient storage and analysis of huge amount of sequence data generated [1]. Despite recent advances in bioinformatics and computational biology, there are many new research problems along with the not yet completely solved problems such as multiple sequence alignment, structure and function prediction, and molecular clustering and classification. Machine learning is a promising approach, with a rich collection of algorithms having strong mathematical basis [2].

1.1 Machine learning

Machine learning is a subfield of artificial intelligence and is concerned with the development of algorithms that allow computers to learn. Machine learning is a process of programming computers using previous experience or training data to increase the

speed and efficiency of the system. The idea behind these techniques is to learn the theory automatically from the data, through a process of inference, model fitting or learning from examples. A model is defined with some parameters and learning is the implementation of a computer program using example data to optimize the execution of the model. In future the model can make predictions with the testing data using the knowledge attained through learning from examples [3]. Learning process is required in cases where incorporating past experiences is needed to solve the problem. Also in cases where the problem is to analyze data which changes from time to time and when systems are required to understand the previous data through learning.

The essence of learning process is drawing logical conclusions from example data. Machine learning builds mathematical models on the basis of statistical learning theory. These models assist the task of making inferences from sample data. The process is composed of two factors: one is implementing efficient algorithms in training to solve the training data and the other is representation of learned model for further analysis of testing data. The domain of machine learning includes supervised learning, unsupervised learning and reinforcement learning. Supervised learning is the process of learning a function with training data having pairs of input objects and desired outputs. Unsupervised learning process deals with training data having only input instances without associated outputs. Reinforcement learning is closer to supervised learning which receives feedback about the correctness of result produced [4].

Machine learning approaches are used in many applications such as pattern recognition, search engines, natural language processing, and stock market analysis etc. Current research domains in Bioinformatics where machine learning techniques applied are

multiple sequence alignment, structure and function prediction, molecular clustering and classification, and expression analysis [5]. Machine learning has received great success in many applications but the study of systematic ways to employ machine learning is meager. In this research, we tried to find a systematic way for integrating different machine learning techniques in the influenza sequence analysis domain. Recent advances in sequence analysis research coupled with increase in ability to generate high-through biological data create new opportunities for collaboration between two communities and pose new challenges for machine learning techniques. Bioinformatics provides an interesting application domain for machine learning techniques that learn models from data for classification, regression, and prediction problems.

1.2 Influenza

Influenza, normally known as flu, is a contagious disease caused by RNA viruses. Influenza virus belongs to Orthomyoxoviridae family and is made up viral envelope containing types of glycol proteins, enclosed around a central core. The central core consists of viral RNA genome and other proteins. The viral genome contains eight pieces of single stranded, negative sense RNA, which encode up to eleven proteins. The influenza virus comprises of three types: A, B, and C [6]. The type A viruses are the most virulent human pathogens among the three influenza types and causes most severe diseases. Mutations in the antigenic structure of the influenza virus have resulted in a number of different influenza subtypes and strains. So far, there are 16 H and 9 N serotypes found in influenza A virus; and different strains of H1N1 and H3N2 circulate in humans during annual influenza season [7].

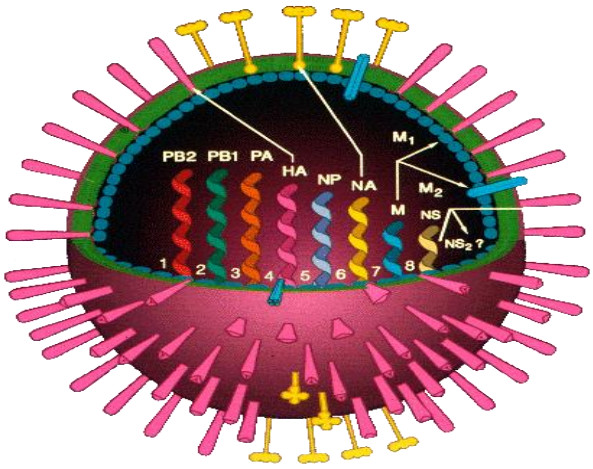

Fig.1.1. Genetic Structure of Influenza virus

An influenza pandemic occurs when a new influenza virus appears against which the human population has no immunity. In the past, influenza pandemics have resulted in increased death, disease and great social disruption. Influenza A virus caused three major global epidemics during the 20th century: the Spanish Flu in 1918, Asian Flu in 1957 and Hong Kong Flu in 1968 [8]. These pandemics were caused by strains of influenza A virus that had undergone major genetic changes and for which the population did not possess significant immunity. The 2009 flu pandemic is a global outbreak of a new influenza A virus H1N1 strain, identified in April 2009 and commonly referred to as Swine Flu [9]. Analysis suggests that the H1N1 strain responsible for the current outbreak first evolve around September 2008 and circulate in the human population for several months before the first cases were identified [10].

The changes in influenza viruses occur in two ways: antigenic drift and antigenic shift [11]. The changes due to antigenic drift happen all the time, but changes due to antigenic shift occurs only occasionally.

Antigenic drift refers to small, gradual changes that occur in the surface proteins HA and

N. These variations are within the HN subtype and changes from time to time. Antigenic drift allows the virus to elude some host immunity. Antigenic drift has been responsible for heavier-than-normal flu seasons.

Antigenic shift refers to variations between subtypes. It is a process by which two different strains of influenza combine to form a new subtype having a mixture genes derived from the parental strains. Because the human immune system has difficulty recognizing the new influenza strain, they are highly dangerous and spread easily from person to person resulting in pandemics.

1.3 Problem Definition and Motivation

The exponential growth in genome sequence data over recent years has led to better understanding of influenza's capacity to target different hosts and its high mutation rate [12]. However, there are several questions need to be solved regarding the seasonal attack of influenza virus and the patterns behind the virus's evolution. Influenza pandemic control requires two steps: first, early detection of new pandemic variant of influenza virus and development of vaccines. It requires a system for rapid detection of pandemic strains which involves: early detection of pandemic viral strains, identifying the origin of viral strains, and classification of virus strains from different subtypes and from different hosts of origin. Identification of influenza virus requires prediction of the subtype of the strain and the host of origin.

Influenza comes back every year with slight changes and the vaccine has to be updated regularly. Identifying the outbreak strains and predicting the influenza lineage helps in developing influenza countermeasures such as virus-specific vaccines and diagnostic

tools. Influenza A viruses can be identified based upon the antigenic properties of their HA and NA [13-14]. Influenza's subtype can be found through a BLAST search [15]. However, there are issues associated with these methods as described in Section 3.1 (Review of current literature).

A number of machine learning techniques are applied to identify relationships or associations in biological data, to group similar genetic elements, to analyze and predict diseases. Machine learning is concerned with the automatic acquisition of models from data, and using such models for automatic inference and prediction. As addressing influenza viral analysis is concerned with classifying viral strains, identifying the role of specific positions and modeling them for future prediction, machine learning techniques have much to offer. Biological data contains large number of features, complex relationships and often lacks clear theory behind it. Machine learning works well with this data because of their capabilities in handling randomness, uncertainty of data noise and in generalization. In order to effectively deal with the problem of influenza sequence analysis, we have explored approaches using machine learning techniques.

The focus of the thesis is directed towards effective utilization of machine learning techniques for identifying subtle differences in the influenza A virus data through classification and developing a prediction tool for detecting the influenza viral strains. The goal of the work presented is to introduce an integrated approach with decision tree and hidden Markov model methods, and a feed-forward backpropagation neural network. Support vector machine has also been employed to provide a comparison of classification results from machine learning techniques. Using machine learning methods to predict the

lineage of influenza virus is much cheaper and faster, yet usually can still yield high accuracy.

1.4 Bioinformatics terminology

The terminology pertaining to biology virus data is explained below

1.4.1 Segment

Influenza A virus genome is contained on eight single RNA strands that code for eleven proteins. The segmented nature of the genome allows the exchange of entire genes between different viral strains. The eight RNA segments are HA (hemagglutinin), NA (neuraminidase), NP (nucleoprotein), M (two matrix proteins, M1 and M2), NS (two distinct non-structural proteins, NS1 and NEP), PA (RNA polymerase), PB1 (RNA polymerase and PB1-F2 protein), and PB2 (RNA polymerase) [16].

1.4.2 Subtypes & Strains

Influenza A viruses are divided into subtypes based on antibody responses to the proteins on the surface of the virus. These surface proteins are called Hemagglutinin (HA) and Neuraminidase (NA), which form the basis of H and N distinctions [17]. Totally, there are 16 different HA subtypes and 9 NA subtypes known, but only H 1, 2 and 3, and N 1 and 2 are commonly found in Humans. Subtypes of influenza A virus are characterized into strains. New strains of influenza viruses appear and replace older strains through antigenic drift process.

1.4.3 DNA Sequence

A DNA sequence or nucleotide sequence is a succession of letters representing the primary structure of a DNA molecule strand. Letters A, C, G, and T represent the four

nucleotide bases of a DNA strand – adenine, cytosine, guanine, and thymine. These sequences are derived from the biological raw material through a process called DNA sequencing.

1.4.4 Protein sequence

Amino acid or protein sequence is an arrangement of amino acids in a protein. Proteins can be made from 20 different kinds of amino acids. The structure and function of each protein is determined by the kinds of amino acids used to make it and their arrangement.

1.4.5 Multiple sequence alignment

A sequence alignment is a way of arranging two sequences one by one where the residues under the same column supposed to have a common evolutionary origin [18]. Through the evolutionary relationship, a set of sequences share a lineage and are descended from a common ancestor. This facilitates the classification analysis to perform better. Sequence alignment helps to identify regions of similarity between the sequences. If the same base occurs in all sequences for a column, then this position has been conserved in evolution. If the bases are different then the two bases may be derived from an ancestral base which could be one of the two or neither. Gaps or dashes are inserted between the residues so that identical or similar characters are aligned in successive columns.

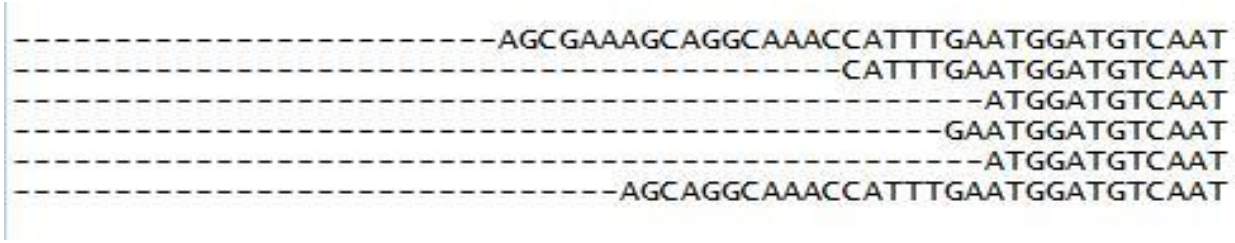

Fig 1.2 A multiple sequence alignment

Multiple sequence alignment is a sequence alignment of three or more sequences. The alignment infers sequence homology and illustrates similarity and dissimilarity among a group of sequences. Fig 1.2 shows a multiple sequence alignment of 7 sequences. Sequences before the alignment may have different lengths but after the alignment all the sequences have same length through insertions and deletions.

We used MUSCLE tool for multiple sequence alignment. MUSCLE stands for multiple sequence comparison by log-expectation, and is one of the most popular multiple alignment software for protein and nucleotide sequences [19]. MUSCLE can achieve both better average accuracy and better speed compared with several other multiple alignment tools such as CLUSTALW [20]-[21] or TCoffee [22], by choosing maximum number of iterations and diagonal optimization.

MUSCLE uses two distance parameters: *k-mer* and Kimura for a pair of sequences. The *K-mer* distance is used for an unaligned pair of sequences and Kimura distance is used for an aligned pair of sequences. A *k-mer* is a contiguous subsequence of length *k*. Sequences having more *k-mer*s in common tend to be similar to each other. For an aligned pair of sequences, the pair wise identity is computed and converted to distance estimate applying Kimura correction for multiple substitutions at a single site. MUSCLE uses UPGMA for clustering distance matrices. A new profile function is used to apply pair wise alignment to profiles.

1.4.6 Phylogenetic tree analysis

Phylogenetic analysis is important to understand the evolution of species and of gene and protein families. Phylogenetic trees define the functional subfamilies within protein or gene families with multiple functions. Neighbor-Joining is a method for reconstructing

phylogenies from a set of distances between each pair of sequences by successive clustering [23]. Neighbor-Joining can reconstruct trees with additive edge lengths without making the assumption that the divergence of the sequences occurs at the same constant rate at all points in the tree.

1.5 Outline of this remaining thesis

The remainder of this thesis is divided into 6 chapters.

Chapter 2 reviews the background theory for this thesis. The background is about the machine learning techniques used for classification analysis.

Chapter 3 explains the current literature and research methodology. This chapter presents a review of the current literature pertaining to influenza analyses and machine learning methods. This chapter reviews the current problems and presents a thesis statement. A brief description about the approach and methodology used is provided.

Chapter 4 describes a novel approach of integrating decision tree and hidden Markov model for the classification and prediction of the influenza A viral subtypes.

Chapter 5 explains the application of machine learning techniques for classifying H1N1 viral strains occurring in 2009 flu pandemic. This chapter reviews the decision tree, support vector machine approach for classification analysis and hidden Markov model for prediction of viral hosts.

Chapter 6 describes the application of neural networks. The architecture of the neural network employed, training and testing models are explained.

Chapter 7 concludes the thesis with a summary of the methods and discusses about the future explorations.

Chapter 2

Background

This chapter presents background of the influenza terminology and machine learning techniques. Section 2.1 presents decision tree method and C 4.5 algorithm which is used to build the decision tree using the tool Weka. Section 2.2 discuss about the support vector machine method. Section 2.3 presents the basics of artificial neural network and the architecture of feed forward back propagation neural network. Section 2.4 explains the probabilistic prediction using hidden Markov model. Section 2.5 provides summary of the topics discussed.

2.1 Decision tree learning

A decision tree is simple yet powerful machine learning algorithm that has been successfully applied for classification problems. Decision tree classification as a standard machine learning technique has been used for a wide range of applications in bioinformatics [24]. The decision tree technique employs a supervised approach for classification, where the leaves on the tree represent classifications and the branches represent association of attributes that result classification. While classifying an instance, a number of decisions are made from root to leaf nodes and the instance is classified to the one of the leaf nodes at the end of the traversal. Each time a value of given instance is compared to the decision function at the node to decide which branch to follow. Finally a decision tree is constructed using the training data by reducing the depth of each path from root to leaf node.

Decision tree uses a tree-like graph and consists of Nodes (Internal and Leaf) and

Branches, where

- Internal node represents a test on an attribute
- Branch represents the result of the test
- Leaf node represents classes or categories

A simple decision tree is a tree depiction of set of rules. A sample decision tree and training set is given below

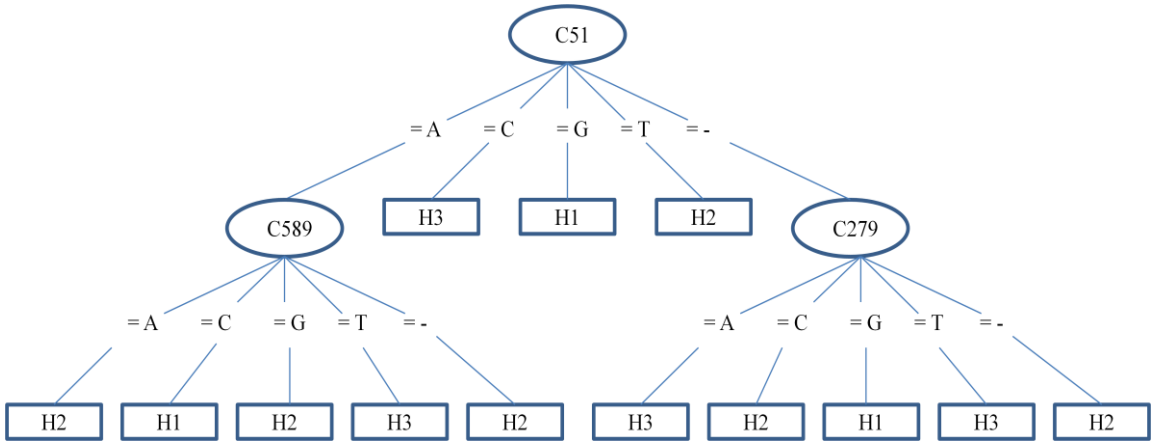

Figure 2.1 Sample decision tree

The decision tree classifies three classes: H1, H2, and H3. The C51, C279, and C589 are attribute representations for the positions 51, 279, and 589. The values for these attributes are nucleotide bases (A, C, G, and T). The decision tree tries to find the answer by using the attributes as positions in the sequence. The decision tree can be represented as a set of Boolean expressions or rules for H1 and can be written as

$$(C51 = G) \vee$$

$$(C51 = A \wedge C589 = C) \vee$$

$$(C51 = - \wedge C279 = G)$$

All the rules for the decision tree above are shown in Table 2.1. The attribute value NA (not available) represents the respective attribute is not used in that rule.

Attributes			Class
C51	**C279**	**C589**	
A	A	NA	H2
A	C	NA	H1
A	G	NA	H2
A	T	NA	H3
A	-	NA	H2
C	NA	NA	H3
G	NA	NA	H1
T	NA	NA	H2
-	NA	A	H3
-	NA	C	H2
-	NA	G	H1
-	NA	T	H3
-	NA	-	H2

Table 2.1 Table representing the sample decision tree

C4.5 is a population algorithm to generate a decision tree. It is also known as statistical classifier as the decision trees generated by the algorithm can be used for classification [25]. It is an extension to the ID3 algorithm, both invented by Ross Quinlan. C4.5 ameliorates the ID3 method by introducing range selection and reduced error pruning. The training data is a set of samples

where each sample is a vector of attributes or features. The algorithm builds trees from the training data with the help of information entropy.

2.1.1 Information Gain and Entropy Measure

Information gain is used to determine how well an attribute separates the training data according to the target concept. It is based on a measure commonly used in information theory known as entropy. Defined over a collection of training data, S, with a Boolean target concept, the entropy of S is defined as:

$$Entropy(S) = -p_{(+)} \log_2 p_{(+)} - p_{(-)} \log_2 p_{(-)}$$

where $p_{(+)}$ is the proportion of positive examples in S and $p_{(-)}$ the proportion of negative examples.

Information gain is the expected reduction in entropy when partitioning the examples of a set S, according to an attribute A. It is defined as:

$$Gain(S, A) = Entropy(S) - \sum_{v \in Values(A)} \frac{|S_v|}{|S|} Entropy(S_v)$$

where Values(A) is the set of all possible values for an attribute A and S_v is the subset of examples in S which have the value v for attribute A.

An attribute is chosen at each node that effectively splits the data. It uses the normalized information gain as a measure for the attribute selection. The attribute with the highest gain is chosen to make the decision.

The time complexity of the algorithm is $O(mn \log n) + O(n \log n)^2$, where n is the number of instances and m is the number of attributes for each instance. The first term in

the complexity denotes the time to reach a depth of $log\, n$, considering all instances and attributes. Hence the total time to reach the depth is $mn\, log\, n$. At each node, error estimation is made and is considered for replacement. At each level of the tree, all the sub trees may be reclassified and the computational cost for this step is $O(nlog\, n)^2$. C4.5 is very efficient and is considered as benchmark for speed and accuracy.

2.1.2 Overfitting and decision trees

Overfitting is not a problem that is inherent to decision tree learners alone. It can occur with any learning algorithm that encounters noisy data or data in which one class, or both classes, are underrepresented. A decision tree, or any learned hypothesis h, is said to overfit training data if there exists another hypothesis h′ that has a larger error than h when tested on the training data, but a smaller error than h when tested on the entire data set. At this point the discussion of overfitting will focus on the extension of ID3 that is used by decision trees algorithms such as C4.5 and C5.0 in an attempt and avoid overfitting data.

There are two common approaches that decision tree induction algorithms can use to avoid overfitting training data. They are:

- Stop the training algorithm before it reaches a point in which it perfectly fits the training data, and,
- Prune the induced decision tree.

The most commonly used is the second approach [26]. Decision tree learners normally employ post-pruning techniques that evaluate the performance of decision trees as they are pruned using a validation set of examples that are not used during training. The goal of pruning is to improve a learner's accuracy on the validation set of data.

In its simplest form post-pruning operates by considering each node in the decision tree as a candidate for pruning. Any node can be removed and assigned the most common class of the training examples that are sorted to the node in question. A node is pruned, if removing it does not make the decision tree perform any worse on the validation set than before the node was removed. By using a validation set of examples it is hoped that the regularities in the data used for training do not occur in the validation set. In this way pruning nodes created on regularities occurring in the training data will not hurt the performance of the decision tree over the validation set.

Pruning techniques do not always use additional data such as the following pruning technique used by C4.5.

C4.5 begins pruning by taking a decision tree to be and converting it into a set of rules; one for each path from the root node to a leaf. Each rule is then generalized by removing any of its conditions that will improve the estimated accuracy of the rule. The rules are then sorted by this estimated accuracy and are considered in the sorted sequence when classifying newly encountered examples. The estimated accuracy of each rule is calculated on the training data used to create the classifier (i.e., it is a measure of how well the rule classifies the training examples). The estimate is a pessimistic one and is calculated by taking the accuracy of the rule over the training examples it covers and then calculating the standard deviation assuming a binomial distribution. For a given confidence level, the lower-bound estimate is taken as a measure of the rules performance. A more detailed discussion of C4.5's pruning technique can be found in [27].

2.2 Support vector machine

The Support vector machines (SVM) is one of the most popular approaches to classification and supervised learning. SVMs are a group of supervised learning methods. SVM theory is based on the concept of decision planes where the training data is represented in multidimensional space and separated by a plane or hyper plane defining two or more classes of data [28]. SVM uses the idea of kernel substitution and belong to the family of linear classifiers. SVM supports both regression and classification tasks and can handle multiple continuous and categorical variables [29].

SVM achieves high generalization performance through structural behavior based on the principle of structural risk minimization and reduces the over fitting problem [30]. The basic SVM can only handle two class classifications.

The two important parameters in theory are w the weight vector, b the bias. The novelty of SVMs lies in choosing the hyper plane $w.p + b = 0$ by maximizing the distance between the nearest data point p_i on each side to the hyper plane.

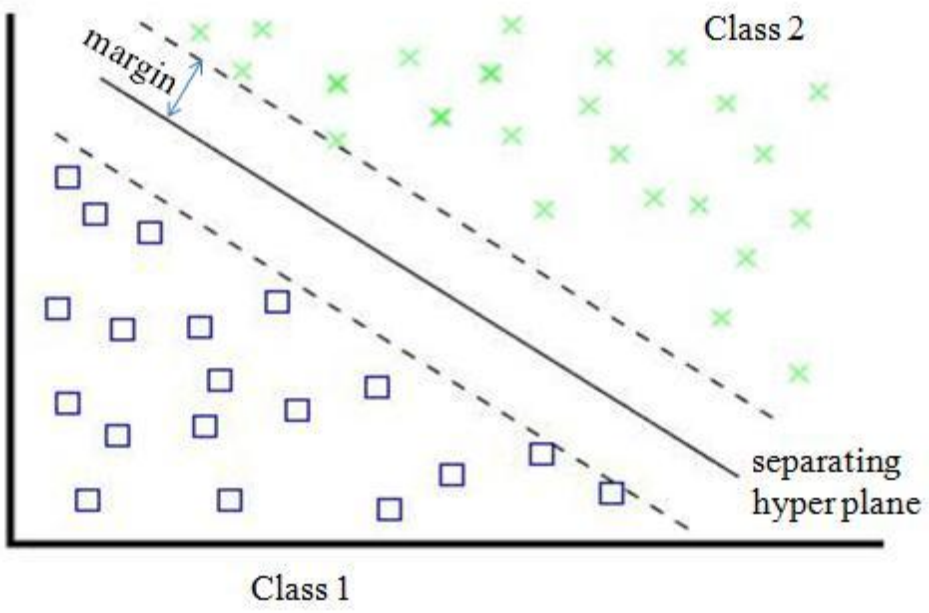

Figure 2.2 Sample SVM classifications

The distance between the data point and hyper plane is known as margin. Hence the point of interest is to maximize the margin. This is done by solving

$$max_{(w,b)} \left(min_i \, d(\pi_{w,b} , p_i) \right)$$

where $d(\pi_{w,b} , p_i) = {|w.p_i + b|}/{||w||}$ is the distance between the data point p_i to the plane π_w

The hyper plane achieving maximized margin is known as maximum margin hyper plane and the classifier is known as maximum margin classifier.

SVM performs discriminative classification, which learns by example to predict the classifications of previously unclassified data. It has strong theoretical foundation based on

- Structural Risk Minimization (SRM): SRM adjusts the training data and balances

the complexity of the model to addresses the problem of over fitting.

- Vapnik-Chervonenkis (VC) Dimension: VC dimension is a measure of the capacity of a statistical classification algorithm. It is used to predict the probabilistic upper bound on the test error of a classification model.

SVM is an alignment-free method and does not depend on multiple alignment, thus it can avoid errors, if any, in multiple alignment files. Another advantage of SVM is that it can classify sequences with low similarity and/or even with very short lengths.

2.3 Artificial neural network

A neural network is a strong data modeling tool that is able to capture and represent complex input-output relationships. It is an interconnected assembly of simple processing elements called units, whose functionality is based on the human neuron. The processing power of the network is stored in the connection weights between the units. Artificial neural network draw much of their inspiration from biological nervous system [31]. They reproduce many of the concepts and features from biological systems.

2.3.1 Biological inspiration

Neural Network simulates the learning function of biological neural systems by modeling the neuron structure of the human brain [32]. The human brain consists of very large number of neurons, interconnected together. The neuron contains a branching input structure called dendrites, a cell body, and a branching output structure known as axon. The axons of one cell are connected to dendrites of another cell via a synapse. Synapses are electrochemical contact between the neurons. These neurons process small amounts of information and may or may not activate based on the output of the neuron and preset

threshold level. If a neuron is activated, it fires an electro signal along the axon which transmits the signal as input to other neurons.

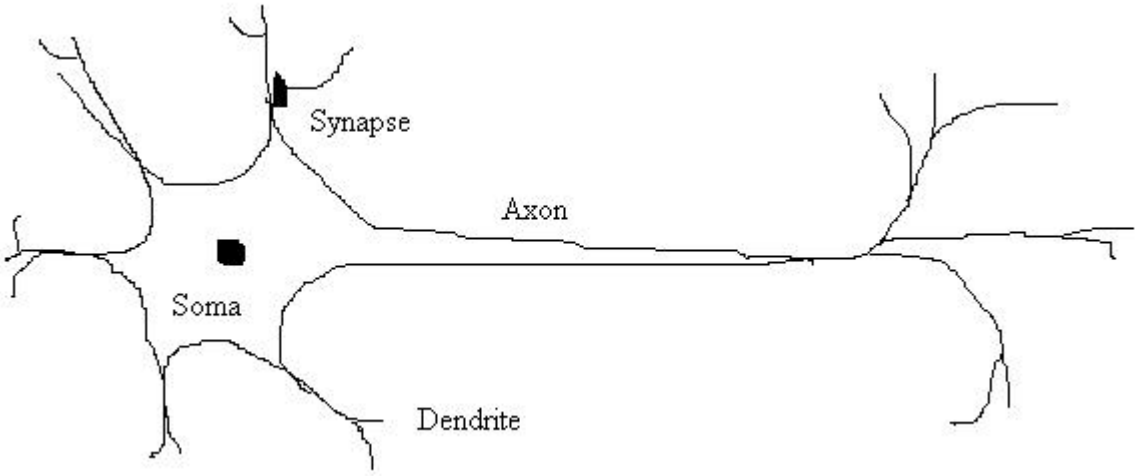

Figure 2.3 A Multi preceptor neuron

2.3.2 Artificial model

To captivate the idea of biological neural systems, an artificial neural network consists of processing units, connections, activation function, and learning algorithm.

Processing units

The processing units are analogous to biological neuron. Processing units have binary interconnections unlike biological neurons with chemical interconnections. Each unit receives a number of inputs either from original data or from the output of other neurons and compute an output signal. Based on the threshold limit, the unit activates to propagate the output signal to other units.

Connections

The units are interconnected to each other by weighted connections. The strength of the connection between two units is known as weight of the connection. The weights are

estimated by learning from training data. These weights play an important role in computing the output signal.

Activation function

The computed signal is passed through an activation function to produce the output of the neuron. The activation function controls the output, such that it will be in certain range of values (0 and 1, or -1 and 1) [33].

Most commonly used activation functions are step (1), sign (2), sigmoid (3) and hyperbolic tangent (4).

$$a(x) = \begin{cases} 1 \; if \; x > 0 \\ 0 \; if \; x \leq 0 \end{cases} \quad (1)$$

$$a(x) = \begin{cases} 1 \; if \; x \geq threshold \\ 0 \; if \; x < threshold \end{cases} \quad (2)$$

$$a(x) = \frac{1}{1 + e^{-x}} \quad (3)$$

$$a(x) = \frac{e^{2x} - 1}{e^{2x} + 1} \quad (4)$$

Step function is like threshold function, which takes a value of 0 if the net input is less than a threshold value, and the value 1 if the net input is greater than the threshold value. Sign function takes a value of -1 if the net input is a negative value (i.e. less than 0) and 1 if the net input is greater than or equal to 0. Sigmoid and tangent are both differentiable functions. The output of the hyperbolic tangent ranges from -1 to 1 and the output from sigmoid ranges from 0 to 1.

Learning

Learning in artificial neural networks occurs during the training phase. After the training phase, the network enters the execution phase where it computes the results for testing data. Learning rule is used to train the neural network by specifying how to adjust the weights for a given input output pair. The learning situations can be categorized into supervised, unsupervised and a hybrid of the two [34]. In supervised learning, the network is provided with input and matching output patterns. In unsupervised learning, the networks learn on their own to recognize patterns in the dataset. The hybrid method combines both supervised and unsupervised learning by learning on their own and then getting feedback response from the environment.

The learning algorithm is a kind of mathematical process which modifies the weights of the connections during each cycle. Backpropagation algorithm is one of the most popular methods for training the neural network, applies a supervised error correction learning method [35].

2.3.3 Feed-forward structure

A simple network with feed forward structure: signals flow from inputs, forwards through any hidden units, eventually reaching the output units. Networks of this type can be trained to provide a desired response to a given input. Learning through feed forward networks is supervised learning [36]. The neurons are arranged in a distinct layered topology. The input layer just presents the values of the input variables. The neurons of hidden and output layers are connected to all of the neurons in the preceding layer. Fig 2.4 shows a fully connected multi layered feed forward neural network with an input

layer, two hidden layers, and an output layer. Although the presented one is fully connected, networks do not need to be fully connected.

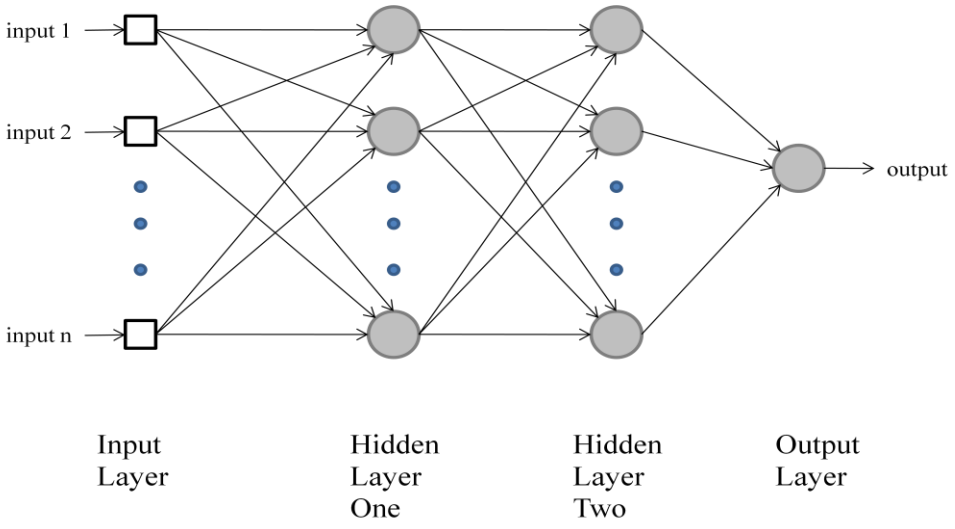

Figure 2.4 A multilayer feed forward ANN

Multilayer feed forward artificial neural networks have two difference phases: A learning phase or training phase and an execution phase. In the training phase the artificial neural network is trained to return a specific output when given a specific input, this is done by continuous training on a set of training data. In the execution phase the artificial neural network returns outputs on the basis of inputs.

When the network is used, the input variable values are placed in the input units, and then the hidden and output layer units are gradually executed. The activation value of each neuron is calculated by taking the weighted sum of the outputs of the units in the preceding layer, and subtracting the threshold.

$$o(N) = a\left(\sum_{i=0}^{n} w_i N_i\right)$$

Where N is a neuron with n inputs (N_0 … N_n) and o is the output from the neuron and w is the weights of the input links (w_0 … w_n). a is an activation function that weights how

powerful the output should be from the neuron, based on the sum of the input. The output from the activation function is a simple threshold either between 0 and 1 or between -1 and 1.

When the entire network has been executed, the outputs of the output layer act as the output of the entire network.

2.3.4 Backpropagation algorithm

Backpropagation is a common method of training artificial neural networks on how to perform a given task [37]. After propagating an input through the network, the error is calculated and the error is propagated back through the network while the weights are adjusted in order to make the error smaller. The most efficient way of minimizing the mean square data for all the training data with the Backpropagation algorithm, is to train on data sequentially one input at a time, instead of training on the combined data.

The Backpropagation algorithm is explained through the following steps

1. After the input is propagated through the artificial neural network to the output, the error e_n on a single output neuron n is calculated as:

   $$e_n = d_n - o_n$$

   Where o_n is the calculated output and d_n is the desired output of neuron n.

2. The error value calculated in the above step is used to calculate a δ_n value, which is used for adjusting the weights. The formula to calculate the adjust value is

   $$\delta_n = e_n \, a^{|}(o_n)$$

   where $a^{|}$ is the derived activation function.

3. The δ_m values for preceding layers can be calculated using the δ_n value from the above step. The equation is

$$\delta_m = \eta\, a'(o_m) \sum_{n=0}^{N} \delta_n\, w_{nm}$$

Where N is the number of neurons in this layer and η is the learning rate parameter, which determines how much the weight, should be adjusted.

4. By using the δ values, the Δw values that the weights should be adjusted by, can be calculated by

$$\Delta w_{nm} = \delta_n o_m$$

The Δw_{nm} value is used to adjust the weight $w_{nm} = w_{nm} + \Delta w_{nm}$

The Backpropagation algorithm moves on to the next input and adjusts the weights according to the output. This process is continued until a stop threshold is reached. The stop threshold is determined by measuring the mean square error of the training data while training with the training data, with this mean square error reaches a certain threshold, the training is stopped.

2.4 Profile hidden Markov modeling

In recent past HMM techniques are introduced to Bioinformatics problems, which have been used in speech recognition for years. Profile can be viewed as statistical descriptions of the consensus of a multiple sequence alignment. They use position-specific scores for nucleotides and position specific penalties for opening and extending an insertion or deletion. This property of profiles captures important information about the degree of conservation at various positions in the multiple alignments, and the varying degree to

which gaps and insertions are permitted. HMMs have a formal probabilistic basis, which is their advantage over other methods [38].

2.4.1 Modeling using HMM

The Hidden Markov Model (HMM) is used for modeling the informative positions generated from the decision trees. An HMM profile includes more flexible information on a given set of sequences than a single sequence. Therefore, database search methods using profiles is more sensitive to remote similarities than those based on pair wise alignments (e.g., regular BLAST).

HMM profiles were built based upon the most informative sites determined by the decision tree method. To take advantage of most informative sites found through decision tree analysis, we built different HMM profiles for the prediction of each influenza A virus host. HMM profiles are statistical representation of a specific group of informative sites. These profiles are used in the Web prediction program to determine the host of a viral strain through sequence comparison.

2.4.2 Markov chains

Markov chains are a series of states with probabilities associated with each transition between states. These probabilities computed from the current state are independent of its previous states.

A hidden Markov model is defined by specifying the following things

- S is the set of states $\{s_1, s_{2, \dots}, s_n\}$
- O is the output set $\{ o_1, o_{2, \dots}, o_m\}$
- $\pi(i)$ is the probability of being in state s_i at time t = 0

- A = Transition probabilities $= \{a_{ij}\}$
- B = Output Probabilities = $\{b_j(k)\}$

2.4.3 Building HMM profiles

This section describes building a profile HMM model. Let's consider the sequences be

ACGT

ATAG

ACAC

ATGT

The trivial HMM is shown in Fig 2.5.

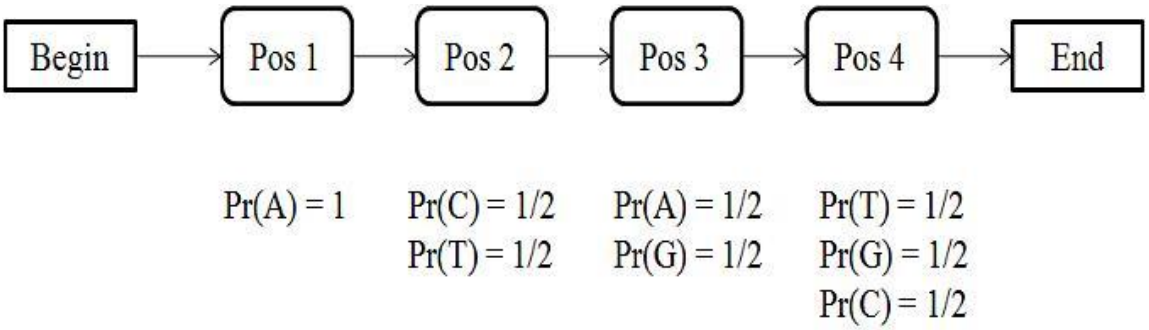

Fig 2.5 A profile HMM

The begin and end states are dummy states which emit no output symbols. The simple profile HMM shown above consists of only match states. Standard profile HMM consists of three types of states, match, insertion and deletion. Insertions are portion of sequences that do not match anything in the model. Generally the output probabilities for insert states are set equal to the background probabilities. Different probabilities can be set for entering different insert states, and this model the fact that insertions may be less well-

tolerated in certain portions of the alignment. The model with the insertions is shown in the Fig 2.6.

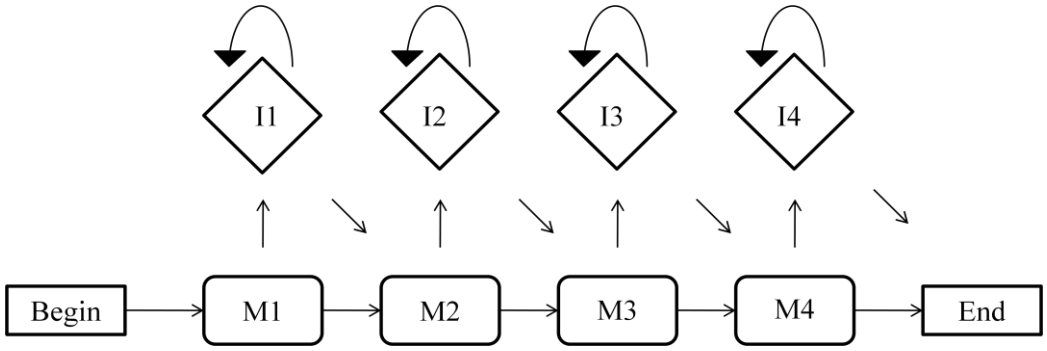

Fig 2.6 Insert states

Long gaps introduce lots of transitions in the model. To avoid this, delete states are modeled which do not emit any symbols. Possible deletions for the above model are shown in the Fig 2.7.

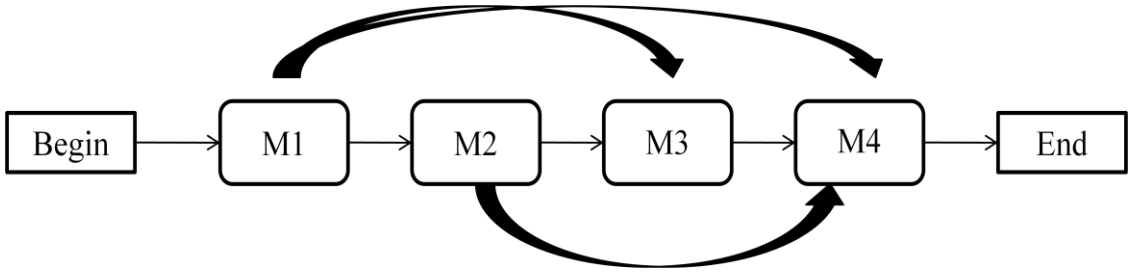

Fig 2.7 Possible deletions

Fig 2.8 gives the model with deletions.

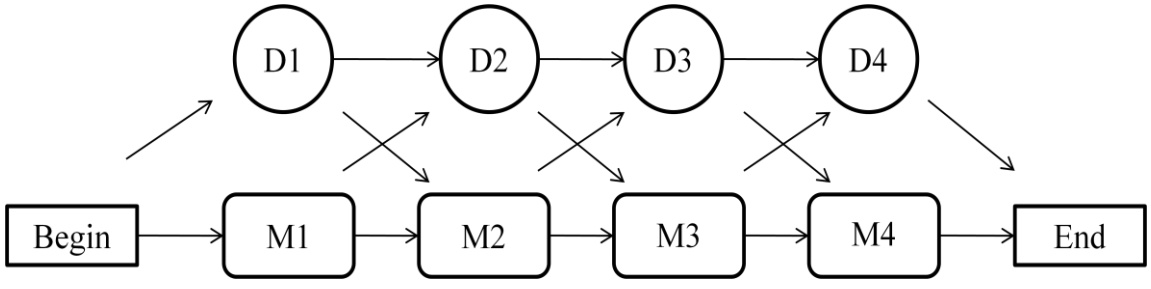

Fig 2.8 Deletions

The overall profile HMM model with insertions and deletions is given below.

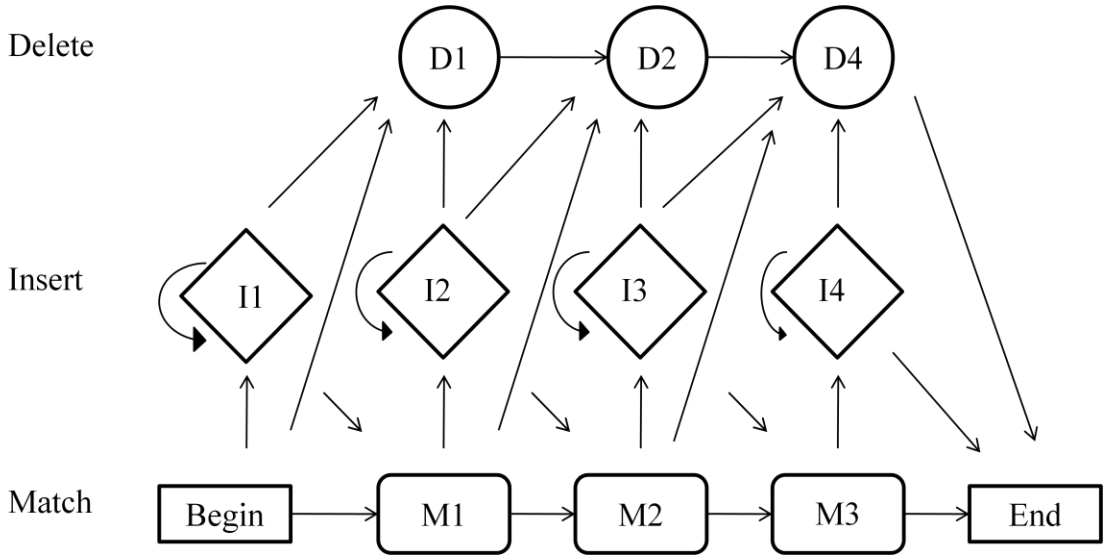

Fig 2.9 Complete HMM

2.4.4 HMMER

HMMER, a package that uses Hidden Markov models (HMMs) for sequence database searching, is used to build HMM models [39]. Profile HMMs can be built using hmmbuild option. Prediction of a new sequence is done by searching the sequence against the profile database. Based on the matching score and e-value the given sequence is predicted. E-value measures the significance of database match. Score is a probability of the given sequence to be related to the model.

2.5 Summary

In this chapter, various machine learning techniques are presented. Learning through decision tree, support vector machine, and neural networks is described. Modeling through hidden Markov models is discussed.

Chapter 3

Methodology

This aim of this chapter is to present the problems associated with the influenza analysis and to explain the research methodology applied. A review of current literature pertaining to machine learning and influenza research is presented. The research statement and the methodology employed are explained. Finally, an experiment design of the thesis is presented.

Section 3.1 reviews the current literature related to machine learning approaches and influenza research. Section 3.2 presents the thesis statement. Section 3.3 presents the experiment design. Section 3.4 introduces an overview of the research methodology.

3.1 Review of current literature

This section reviews the current literature. The literature reviewed is organized into two categories. The first category reviews research related to influenza analysis and the second category reviews research related to machine learning techniques.

3.1.1 Influenza research

Traditionally the analysis of influenza virus evolution relied on phylogenetic techniques. Phylogenetic trees provide an insight into virus origins but cannot provide the subtle differences between different classifications of influenza features. The phylogenetic trees classify the sequences by defining function subfamilies and describe the evolutionary relationships. Song et al. described the phylogenetic analysis of H1N1 swine influenza virus isolation in Korea [40]. However, inferring phylogenetic trees is difficult and these

trees do not show a clear picture of different clusters. Phylogenetic trees do not have mathematical origins and may not always represent the accurate evolutionary relationships.

Experimentally, influenza A viruses can be identified based upon their antigenic properties of the Hemagglutinin (H) and neuraminidase (N) glycol proteins expressed on the surface of virus particles. The classical ways of determining the subtype of influenza virus for HA and NA segments are hemagglutination inhibition (HI) assay and neuraminidase-inhibition (NI) assay described by Pedersen [41]. They are capable of distinguishing antigenic differences between influenza even of the same subtype. This antigenic assay, however, has several disadvantages; including 1) the development of antisera and antigens is very time-consuming. 2) The outcome is heavily dependent on the quality of the reagents used and 3) the assay provides qualitative rather than quantitative information [42]. 4) When working with uncharacterized viruses or antibody subtypes, the library of reference reagents required for identifying antigentically distinct influenza viruses and/or antibody specificities from multiple lineages of a single Hemagglutinin subtype requires extensive laboratory support for the production and optimization of reagents.

Fouchier et al. [43] described the characterization of influenza novel influenza A virus Hemagglutinin subtype using phylogenetic tree analysis and nucleotide sequence analysis. They generated DNA maximum likelihood trees and achieved accuracies in the range of 86-100% for characterizing subtype of novel influenza sequences. The trees are to be generated every time when a new sequence is characterized and they do not provide critical information which can be used for other data.

Alternatively, sequence analysis, a standard technique, is becoming a preferred method for classification [44]. Viseshakul et al. presented described the sequence analysis of influenza A virus [45]. The sequence analysis is performed for only H5N1 avian influenza A virus and no illustrative positions or patterns are identified for further analysis.

Another common way to find which subtype or host, a genetic sequence belongs to is through the BLAST search [46]. However, there are issues associate with the BLAST algorithm when used to identify global sequence similarity as explained in Lu et al [47]. Most importantly, the BLAST result can not reveal important mutations that may be functionally related to the structure and function of proteins.

Mutations are the changes in viral RNA by which influenza viruses' change over time and results in new viral strains. Accurate prediction of future mutations is critical for vaccine development and influenza surveillance. There are no complete methods to trace evolutionary changes that occur over time. General methods such as sequence comparison, phylogenetics, and sequence alignment use amino acids in a protein as alphabet and determine the evolution of influenza virus. These methods lack statistical modeling and cannot accurately predict the mutation changes. Characterization of subtype and host of influenza virus depends mainly on antigenic assays, which is time consuming and not accurate.

3.1.2 Machine learning literature in bioinformatics

Decision tree is able to determine the class of an instance through the values of its attributes. The basic procedure for building decision trees is deterministic. Most prior research in decision tree learning is centered on construction of best decision trees,

extracting rules for classification problems and to an extent in the framework of probabilistic decision trees. Traditional decision tree learning algorithms such as ID3, CART and C4.5 concentrated more on reducing the size of decision tree and choosing best appropriate branching attributes. Umar Syed and Golan Yona [48] introduced a mixture model of probabilistic decision trees to learn complex relationships in the data. They proposed a probabilistic based method for searching the hypothesis space to address the issues of deterministic models.

Sami, A and Takahashi [49], converted DNA based problems to regular data mining methods and presented a method that can be applied to all classes of classification. Salzberg et al developed a decision tree system for finding genes in DNA [50]. Lee et al. described the application of decision trees for the classification of antimicrobial peptides. Yuan et al compared decision trees and support vector machines in classifying gene expressions [51]. They build a SVM bank with all the possible encoding schemes. They have used nucleotide sequence data with possible coding schemes (00 01 10 11).

The major issues in the implementation of neural networks are network architecture, input/output encoding and training algorithm. Fahlman and Lebiere [52] described a cascade correlation learning architecture. The approach starts with a small network and new units and layers are added gradually to solve the problem. Reed [53] discussed a network trimming approach which starts with a large network enough to determine the problem and then eliminating needless units.

Arthur E. Bryson and Yu-Chi Ho reported the backpropagation algorithm [37]. The algorithm learns by changing weights in a feed forward neural network, with the use of

activation functions. Simon Haykin [54] described the feedforward architecture, which is most commonly used with backpropagation training algorithm.

Brunak et al. [55] described a direct input encoding scheme which uses four units to represent a nucleotide. Demeler and Zhou [56] used a dense representation using two units for four nucleotides and showed that this representation is better than using four units. Wu et al. [57] presented an indirect encoding method which calculates frequencies of individuals or a certain group of bases and using these frequencies as inputs. Farber et al. [58] presented a comparative study between direct and indirect encoding methods and observed that 2-Mer frequency representation method performed better than direct encoding methods.

3.2 Thesis statement

The objective of this thesis is applying machine learning techniques such as decision tree, support vector machine, and neural networks to develop a computational system for classifying influenza virus subtypes and hosts.

In order to develop a rapid system for detecting influenza viral strains, different machine learning techniques were applied to develop classification models for influenza subtypes and hosts. These classification models can be used to detect the host and subtype of a given new influenza sequence.

3.3 Experiment design

The experimental design is summarized in Fig 3.2. In brief, the comparison analysis is conducted with three methods, decision tree, support vector machines, and neural

networks and prediction system is developed using decision tree results with hidden markov modeling.

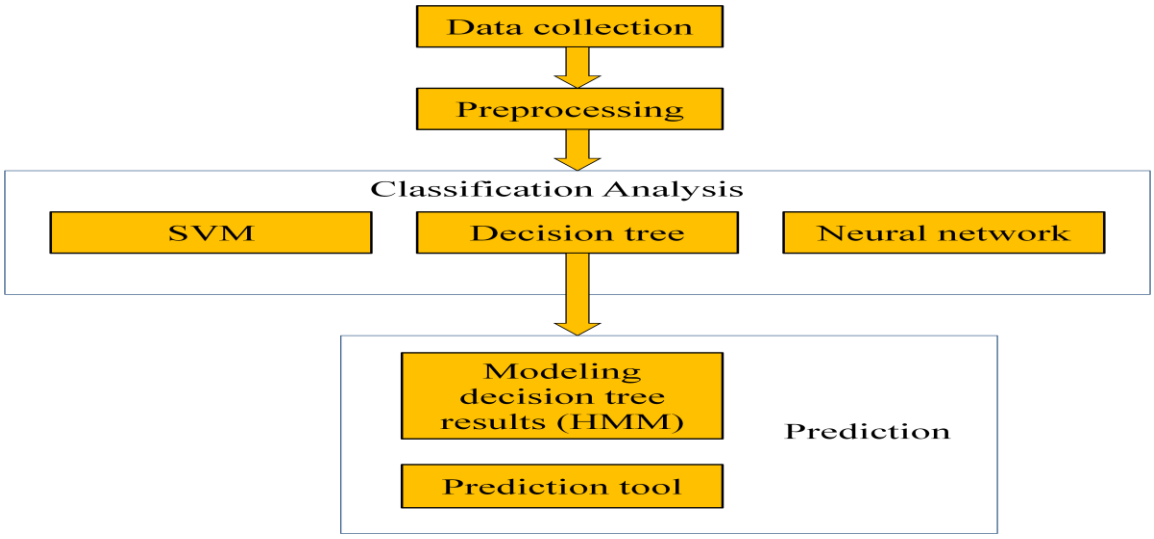

Fig 3.2 Experimental design

3.4 Research methodology

Various machine learning methods have found numerous applications in many fields, including bioinformatics. For example, [Ozen 2009] discussed the issue of machine learning integration for predicting the effect of single amino acid substitutions on protein stability. However, although the importance of integrated play of these methods have long been recognized, in practice the integrated use of the difference methods is still in an ad hoc manner. Systematic investigations for when to use each method and how to combine different methods are needed, which can offer valuable advices to bioinformatics practitioners. Integration of machine learning methods may be achieved at different levels. For example, a tightly coupled machine learning system could be desirable to some end users, but it is hard to achieve due to diversity of the methods and

complexity of real world applications. In our view, a more viable approach would be a methodology which consists of useful guidelines and advices on how to apply domain knowledge for effective use of machine learning methods. This has been our objective of this study.

To achieve effective integrated use of machine learning methods, there are a number of issues to be considered. First, we should have good understanding important features of each individual machine learning method; understand under which conditions a particular method is most applicable. We should also design the overall experimental environment in which different machine learning methods complement to each other. For example, should we employ different methods in parallel, then compare and combine the results together? Or should we apply one particular method first, then past the result to feed into some other method? These and other important questions cannot be answered correctly without the good understanding of the application area. In particular, in the domain of bioinformatics, we need to start with a clearly defined research objective, so that appropriate machine learning methods can be applied to biological objects (such as applying one machine learning method on nucleotide sequences for one particular task and applying some other machine learning method on corresponding amino acid sequences), and combine results obtained from different methods.

In this paper we explore the integrated use of several machine learning methods for influenza analysis. This thesis explores the use of machine learning techniques in developing a computational system for effectively classifying influenza A virus subtypes and hosts. Specifically as shown in the Fig 3.2, decision tree, support vector machine, and

neural network approaches are used for classification and hidden markov modeling method is used for prediction.

The decision tree approach classifies the sequences of different groups based on the nucleotide information at specific positions, whereas the support vector machine approach classifies sequences based on the frequency of amino acids appearing in various sequences. Nucleotide sequences have only 4 bases A, C, G, and T, which is more applicable for decision tree classification as there will be only 4 branches in classification whereas protein sequences have 20 bases. Support Vector Machine combines protein features like localization, amino acid composition, sequential amino acid usage, and whether they are continuous or discrete values, into a feature vector, and take them all into consideration and hence achieve better results using protein data. The neural network considers nucleotide data and learns to map a set of input features, sequence composition, onto a set of binary outputs. This complementary strategy of decision tree and neural network analyses using nucleotide data and support vector machine analysis using protein data provides a comprehensive description of influenza A viral classification

3.4.1 Classification analysis

Three methods were applied for the classification analysis.

Decision tree method uses nucleotide sequence data for the analysis. The sequence data is aligned prior to the analysis and converted to attribute relation file format. Decision tree analysis provides informative positions that lead to classification. Informative positions are the attribute (base) values which classifies different groups (subtypes or hosts) in the decision tree classification. More explanation on informative positions is given in the section 4.3. The main advantage of decision tree method is that the classification rules are

easy to interpret. The informative positions were extracted from classification rules. These positions were used in the prediction system and can be studied further to identify mutations. Several iterations were performed to generate decision trees in order to collect more informative positions. These positions were modeled to develop the prediction system and studied at protein level to identify mutations.

Using decision tree analysis for sequence classification has several advantages. First, the sequence is not classified based on a single position, but on set of positions. In this classification, instead of searching complex patterns, decision trees search for positions and combine them. Decision trees express these positions in a tree path. Second, the hierarchical structure of decision tree enforces an order in the usage of informative positions, i.e. given a new sequence, not all positions should be matched in advance but one position at a time based on specific branch. Third, decision tree is a white box model which expresses the rules in simple Boolean logic. The process of classification and the meaning of classifier can be easily explained using the rules.

Decision tree analysis depends on single positions for classification and do not consider patterns or frequency of a group of nucleotide positions. Neural network analysis is applied to consider the frequencies of nucleotide base groups to take advantage of the patterns. Several advantages of artificial neural networks include handling noisy data and having high adaptability. The distinguishing feature of the neural networks is their ability to learn. The training of a neural network is performed by feeding the inputs into the network and pairing them with the corresponding known outputs. The neural network "finds a way to mathematically map those inputs to known outputs" [59]. The neural network does not stop learning at this point and adapt the weights in response to gradual

changes over time through supervised learning. The relationships among variables are represented as patterns of inter-related data and the neural network searches for patterns during the learning stage. Neural networks have the ability to identify complex relationships between dependent and independent variables and the ability to detect all possible interactions between predictor variables. The strength of the artificial neural network is their ability to find patterns despite missing data. The static and non-linear function used by neural networks provides a method to fit the parameters of a particular function to a given set of data. Neural network analysis is performed without the constraints of requiring a priori model as the basis and can make use of unlimited number of features of the data being analyzed. Neural networks do not require any preconceived notion about what features of data are important. The internal network representation of the data using weights contains the discrimination criteria.

A feed forward network is designed and used a supervised learning backpropagation algorithm for training. The frequencies of different lengths of nucleotide groups were calculated and presented as input to the network. Non aligned nucleotide sequence data is used to calculate frequencies. The frequencies were represented using *k*-mer frequencies. The *k*-mer frequencies are the frequencies of a group of nucleotide bases together in the whole sequence. The length of the group is given by the value of *k.* For example, in a sequence "ACGTAC", the 2-Mer groups are AC, CG, GT, and TA. The frequencies are 0.4 for AC and 0.2 for CG, GT, and TA. Different values of *k* are tested for the classification analysis.

The decision tree and neural network methods are based on nucleotide data and decision tree method depends on multiple sequence alignment. Sequence alignment may cause

errors due to the presence of divergent sequences. Support vector machine is an alignment free approach and uses protein sequence data for the classification analysis. The basic classification SVM creates a maximum-margin hyper plane between classes in a transformed input space. Protein sequence data contains 20 amino acids compared to 4 bases in nucleotide data. To take advantage of the protein features support vector machine method is applied. Support vector machine is a kernel based classifier and can take advantage of useful kernel properties. The goal is to merge features at kernel level before doing the concept of classification. Support vector machine has the ability of handling large feature spaces, effectively avoids over fitting and summarizes information from the data sets. The decision model of the support vector machine reflects the regularities of the training data rather than incapability of the learning machine. The key feature of support vector machine is that some sequences are more important than others and by concentrating more on them guarantees better generalization. Also a support vector machine reduces complexity because the prediction of a new sequence is done only by support vectors and not all observations.

Similar to neural network analysis, support vector machine considers the frequencies of certain length of amino acid groups were calculated and present them as input vectors for the classification.

3.4.2 Integration of decision tree and hidden Markov model for prediction system

The results of the decision tree classification analysis contain the informative positions. Decision tree analysis efficiently extracts positions (attribute values) which classify different hosts or subtypes (classes) from the sequence data but does not have probabilistic basis for using them to predict new sequence data. The nucleotide sequences

have large number of bases and many of the sequences belonging to same host group or subtype group often differ in base value for some informative positions. Hence the prediction accuracy using this system which checks base data at all informative positions has an accuracy of 65%. In order to increase the prediction accuracy and effectively utilize the informative positions profiles were modeled using hidden markov modeling.

The informative positions collected from the decision tree analysis were modeled into profiles using hidden Markov modeling. HMM profiles provide statistical representation of a specific group of informative sites. These profiles are used in the web prediction program to determine the host or subtype of a viral strain through sequence comparison.

3.4.3 Studying informative positions at protein level

The informative positions found the decision tree analysis in classifying different subtypes were collected together. These positions are converted at protein level and compared with the mutation positions found through the literature study. The conversion of these positions to protein level allows the identification of mutations which are responsible for the influenza outbreaks. A mutation database is built to store the mutation positions found through data mining using decision tree analysis and literature study.

3.5 Summary

This chapter reviews the problems associated with the influenza analysis and presents the research methodology. The reasons for applying various classification methods are explained. The idea of integrating the decision tree and hidden Markov modeling is described. The use of informative positions at protein level is provided.

Chapter 4

Integration of decision tree and hidden Markov model for subtype prediction of human influenza A virus

The purpose of this chapter is to describe the integration of decision tree and hidden Markov modeling techniques for subtype prediction and to present the classification results. Influenza A viral sequences were classified through decision tree analysis based on hosts and subtypes. The informative positions generated at each of iteration steps of the classification analysis are collected together for each subtype and hosts. The informative positions are developed into models using hidden Markov modeling. These models are used to build a Web prediction system which detects the subtype and host of a given set of sequences. Furthermore, the informative positions are converted at protein level and compared with the mutation positions found through the literature study. A mutation database is built with the positions found through decision tree analysis and literature review.

Section 4.1 describes the methodology. Section 4.2 provides the description of data collection and preprocessing. Section 4.3 explains decision tree method application and classification results. Section 4.4 describes the hidden Markov modeling of informative positions found through decision tree classification. Section 4.5 presents a Web prediction system developed using the HMM profiles and its performance. Section 4.6 describes the study of informative positions at protein level to find the mutations.

4.1 Methodology

The methodology of the integration of decision tree classification and hidden Markov modeling is briefly described in the following steps. These steps are explained to detail in the following subsections.

1. Influenza A virus sequence data was collected from the NCBI resource and redundant sequences were removed.
2. Preprocessing the data. Sequences were aligned through multiple sequence alignment to arrange the residues of common ancestor in the same column. Alignment inserts gaps in between the residues to align the residues. The aligned nucleotide sequence data in fasta format was converted to attribute relation file format.
3. Decision trees were computed at each of the iterations and the positions which classify different groups were collected together. These positions were termed as informative positions.
4. Residue data representing informative positions was collected from all the input data and modeled through hidden Markov modeling to develop profile models. These models were developed for each subtype.
5. A Web prediction tool was developed to detect the subtype of a given sequence by using profile hidden Markov models.
6. Informative positions were studied at protein level and compared with the mutations found through literature. These positions were added to the mutation database [60].

The Waikato Environment for Knowledge Analysis (Weka), a machine learning suite, was used to build the decision trees. Weka is a collection of various machine learning and data mining algorithms, developed by the University of Waikato in New Zealand [61, 62]. HMMER, a package that uses hidden Markov models (HMMs) for sequence database searching, was used to build HMM models based upon the most informative sites determined by the decision tree method.

4.2 Data collection and preprocessing

4.2.1 Subtype classification data

Humana influenza A viral sequences were downloaded from the Influenza Virus Resources at NCBI (http://www.ncbi.nlm.nih.gov/genomes/FLU/FLU.html) [63]. A total of 4,413 sequences were used for training the decision tree method, including 2,154 H sequences and 2,259 N sequences (Table 4.1). For the purpose of cross validation, sequences collected were automatically partitioned into two datasets, one for training and the other for testing.

Virus Species	Host	Segment	Subtype	No. of Sequences
Influenza virus A	Human	4 (HA)	H1	987
Influenza virus A	Human	4 (HA)	H2	116
Influenza virus A	Human	4 (HA)	H3	1051
Influenza virus A	Human	4 (HA)	N1	1144
Influenza virus A	Human	4 (HA)	N2	1115

Table 4.1 Human influenza A virus sequences used for Subtype classification

Three H subtypes H1, H2, and H3; and 2 N subtypes N1 and N2 are used for analysis. The varying number of sequence data for different subtypes is due to the availability of the non redundant sequences. Nucleotide sequence data was used in the decision tree analysis.

4.2.2 Eliminating redundant data

The sequence data from the public database domains often contains redundant data. In most cases of redundancy, there will be multiple entries with same sequence data and different descriptions. Therefore, only one sequence of these was considered, excluding other sequences. Table 4.1 is shows the number of sequences after the redundant sequences were removed.

4.2.3 Sequence alignment

In this study, for the subtype classification all Hemagglutinin sequences were aligned collectively and then divided into each subtype group for training; the same procedure was applied to the neuraminidase sequences. The process of alignment is explained in section 1.4.3. For the host classification, all H1N1 nucleotide sequences from a specific segment were aligned collectively and then divided into three host groups for training. The parameters –maxiters and –maxmb in MUSCLE were set to 2 and 250 MB, respectively. Because of the large data set, only the first two iterations of the algorithm were performed to compromise between speed and accuracy. As for the user submitted sequences in Web prediction tool, single iteration was performed as the remaining data is already aligned and to show the results quickly.

4.2.4 Converting sequence data to ARFF format

The nucleotide sequence data consists of succession of letters representing the structure of the sequence. There are four nucleotide bases adenine, cytosine, guanine, and thymine, which are represented by letters A, C, G, and T respectively. Also there is a special symbol '-', which represents a gap formed by insertion or deletion. The sequence data for the analysis downloaded from public databases was in FASTA format.

FASTA is a text based format for representing nucleotide or protein sequence data. The format is simple and easy to manipulate using scripting languages. A FASTA file contains multiple sequences where each sequence begins with a single line description, followed by multiple lines of sequence data. The description of the sequence starts with a >' (greater than) symbol to distinguish from original data. Fig 4.2 shows a group of sequences in Fasta format.

```
>EPI178968 | HA | A/Swine/Indiana/1726/1988 | EPI_ISL_30059 | CY039925 | H1N1
taaaagcaaccaaaatgaaggcaatactattagtcttgctatatacatttacagccgcaaatgcagacacattatgtatcggttatcatgcaaataattcaactgacactgttgatacag
tactagaaaagaatgtaacagtaacacactctgttaaccttctagaagacagacataacggaaaactatgtaaactaaggggggtagccccattgcatttgggtaaatgtaacattgcag
gatggctcctgggaaacccagaatgtgaattactattcacagcaagctcatggtcttacattgtggaaacatctaactcagacaatgggacatgttacccaggagatttcatcaattatg
aagagctaagagagcagttgagctcagtgtcatcatttgaaagatttgagatattccccaaggcaagttcatggcccaatcatgaaacgaatagaggtgtgacggcagcatgcccttatg
ctggagcaaacagcttctacagaaatttaatatggctggtaaaaaaaggaaattcatacccaaagctcagcaaatcctatgttaacaataaggagaaggaagtcctcgtgctatggggca
ttcaccatccacctaccagtactgaccaacaaagtctctaccagaatgcagatgcctatgtttttgtggggtcatcaaagtacaacaagaaattcaagccagaaatagcaacaagaccca
aggtgagaggtcaagcagggagaatgaactattactggacactagtagagcctggagacacaataacattcgaagcaactggaaatctagtggtaccaagatatgccttcgcaatgaaaa
gaggttctggatctggtattatcatttcagatacaccagtccacgattgtaatacgacttgtcaaacacccaaaggtgctataaacaccagcctcccatttcagaatatacatccagtca
caattggagaatgtccaaaatatgtcaaaagcacaaaattgagaatggctacaggactaaggaatatcccgtctattcaatctagaggtctgtttggagccattgctggctttattgagg
>EPI31000 | HA | A/Swine/Spain/50047/2003 | EPI_ISL_5203 | CY009892 | H1N1
aaaattaaatcaaccaaaatggaagtaaaactgtttgtattattctgtgcattcactgcactgaaagctgacaccatttgtgtaggctatcatgctaacaattccacagacactgtcgac
acaatactggagaagaatgtgactgttacccattcagttaacttactagaaaacaaccataatggaaaacttttgtagcctgaatggaaaggccccttacaactggggaactgcaacgta
gcaggatggatccttggcaacccagaatgtgacttgttgctcacagcgaattcgtggtcttacataatagagacttcaaattcaaaaaatggagcatgctacccaggagaattcgctgat
tatgaagaattaagggagcagctgagtacagtctcttcatttgaaagatttgaaattttcccaaaagcaacctcatggccaaaccatgatacaaccagaggtaccacagttgcatgctcc
cattctggagccaacagtttttatcggaacttgctatggatagtaaagaaaggaaactcctatcctaagctcagcaagtcatacacaaacaacaaagggaaagaagtgcttgtaatctgg
ggagtgcaccaccctccgactgacagggaccaacagaccctctaccagaataatcacacatatatttcagttggatcatcaaaatactaccaaaggttcacaccagaaatagtagccaga
cctaaagtcagagaacaagcaggcagaatgaattattattggacactgttagatcagggagacaccataacttttgaagccactgggaatttaatagcaccatggcacgcatttgcattg
aataagggctctagttctggaattatgatgtcggatgctcatgttcacaattgcaccacaaagtgccaaactcctcatggggccttgaaaagcaatcttccttttcagaacgtacatccc
atcactattggagaatgccccaaatatgttaaaagcacccaactaagaatggcaacaggattaaggaacatcccctctgttcaatccagaggacttttggggcaattgccggattcatt
>EPI100032 | HA | A/swine/Alberta/56626/03 | EPI_ISL_9878 | DQ280203 | H1N1
atgaaggcaatactattaatattgctgtatacatttacagccacaaatgcagacacactatgtataggttatcatgcaaataattcaactgacactgttgatacagtgctggaaaagaat
gtaacagtaacacattctgttaaccttctagaagacagacataacggaaaactatgcaacctaggggaatagccccattgtacttgggtaattgtaacattgctggatggcttttggga
aacccagaatgtgaattactattcacagtaagctcatggtcctacattgtggaaacatctaactcagacaatgggacatgttaccctggagatttcatcaattatgaagagctgagagag
cagttgagctcaatgtcatcatatgaaagatttgagatattccccaaagaaagttcatggcccaatcatgaaacaaacaggggtgtgactgcagcatgcccctatgctggagcaaacagt
ttctacagaaacttaatatggctggtaaaaaaaggaaattcatacccaaagatcaacaaatcctatgttaataataaagagaaggaagtccttgtgctatggggcattcaccatccacct
accagtgctgaccaacaaagtctctatcagaatgcagatgcctatgtttctgtggagtcatcaaagtacaacaagaaatttaagccagaaatagcaacaagacccaaggtgagaggtcaa
gcagggagaatgaactattactggacattagtagagcctggagacacaataacattcgaagcaactggaaatctagtggtaccaagatatgccttcgcaatgaaaagaggttctggatct
ggtattatcatttcagatacaccagtccacgattgtaatacgacttgtcaaacacccaaaggtgctataaacaccagcctcccatttcagaatatacatccagtcacaattggagaatgt
ccaaaatatgtcaaaagcacaaaattgagaatggctacaggactaaggaatatcccgtctattcaatctagaggtctgtttggagccattgctggctttattgaggggggatggacagga
atgatagatggatggtacggttatcaccatcaaaatgagcagggatcaggatatgcagct
```

Fig 4.1 Multiple sequences in Fasta format

The native data format of Weka is ".arff" (Attribute Relation File Format). Weka also supports CSV (Comma Separated Values) but CSV files do not contain attribute label information, and Weka needs to generate attribute labels which can create incompatibility problems between the training and testing data sets. An ARFF is an ASCII text file describes a list of instances sharing a set of attributes. The sequence data in Fasta format was converted into ARFF format using Format Conversion Tool [Appendix A1].

A simple ARFF format contains mainly two sections, header section and data section. Fig 4.3 shows a sample format. The header section includes title, the name of the relation, and a list of attributes and their types. Attributes here are the number of bases in a sequence and are defined as nominal attributes. Nominal attributes are provided with a list of possible values taken by the attribute. The nominal attribute declaration is defined by

@ATTRIBUTE class {nominal value1, nominal value2 …}

In this case, attributes C1, C2... C1500 are the positions of the bases in a sequence. They can have any of the possible values {A, C, G, or T}. There are three possible classes Human, Swine, and Human_Swine to which the training data belongs.

The data section contains the data related to the attributes. Data declarations take the form:

@ATTRIBUTE Data, Class

Attribute data of each instance is declared followed by the class to which the instance belongs.

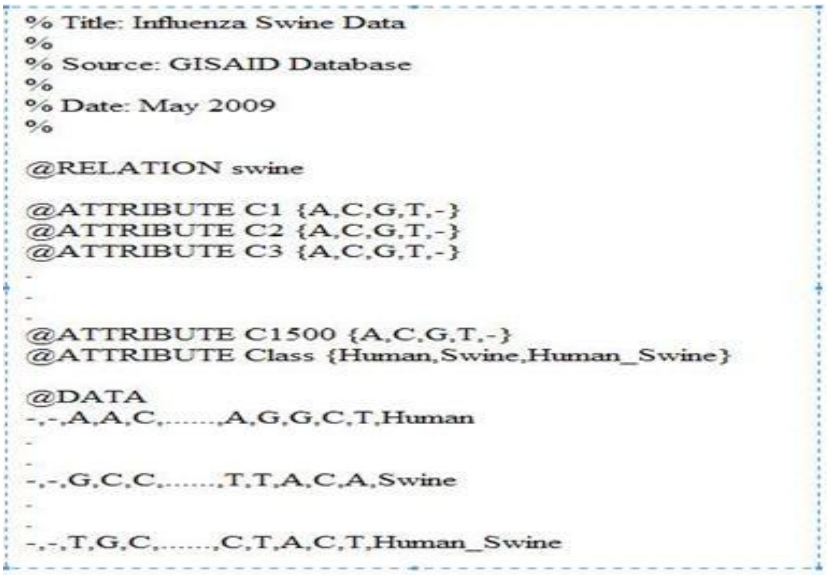

Fig 4.2 ARFF Format

4.3 Decision tree analysis for classifying influenza subtypes

The Weka classifier package has its own version of C4.5 known as J48. The Weka J48 classifier chooses an attribute that best differentiates the output attribute values and creates a separate tree branch for each value of the attribute.

In training, the parameters, confidence factor, maximum number of instances per leaf, number of folds per cross-validation, were set to be 0.25, 2 and 10, respectively. The confidence factor determines the confidence values to be used when pruning the tree. The default value (0.25) works reasonably well in most cases. The maximum number of instances per leaf was set to a lower number (i.e., 2) in order to create a more specialized tree. The 10-fold cross-validation means 90% of the full data is used for training and 10% of the data for testing in each fold. This is breaking the data into 10 sets of size n/10 (where n is the size of the training data provided) and training on 9 data sets and testing

on 1 data set. This process was repeated for 10 times and a mean accuracy was calculated.

The aligned sequences of each type were used to train decision tree classifiers in the J48 program, which allows the most informative nucleotide positions to be found. In each of the iteration steps, one or more critical positions, in which different subtypes can be most likely identified, were determined. These positions were collectively utilized to build HMMs for further subtype prediction.

An example decision tree is shown in Fig. 4.4, where the informative positions C51 and C279 can be used to classify subtypes of HA antigen.

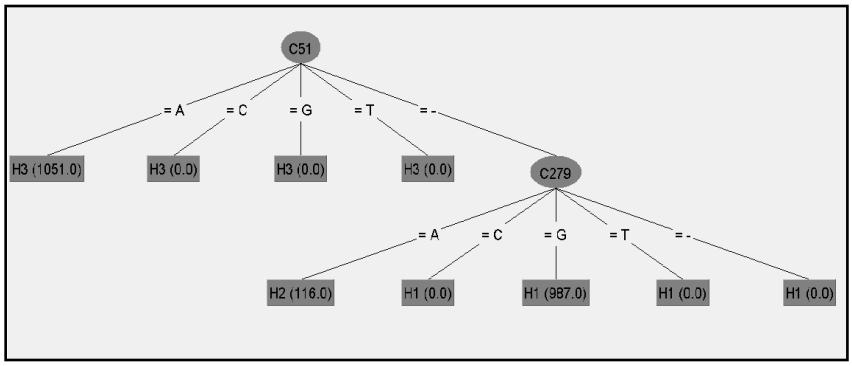

Fig 4.3 A decision tree with informative positions classifying three subtypes, H1, H2 and H3

A total of 28 iterations were performed for H and N subtypes. At each of the iteration steps, a decision tree was generated with one or more informative positions which classify three sets of data (i.e. H1, H2, and H3) for H serotype and two sets of data (i.e. N1 and N2) for N serotype. The accuracies of the classification were in the range of 95-98% for all the iterations. Nucleotide sequences contain 1500 to 2000 bases and many positions exists which can classify sequences between different groups. These positions were called informative positions. In this case, a single decision tree generated during

each iteration identifies only 2-3 positions as informative positions. In order to collect more number of informative positions we performed 28 iterations. After 28 iterations, the results for classification have shown a drop off in accuracy and hence the decision tree generation is stopped.

Table 4.2 summarizes the average accuracy of classification between H1, H2, and H3; N1 and N2 subtypes. Totally 78 most informative positions were found to collectively identify H subtypes; and 63 informative positions were identified for N subtypes as summarized in Table 4.3. These positions determined by decision tree for classifying different groups of data are referred to as informative positions.

Serotype	Accuracy
H	97.16%
N	96.72%

Table 4.2 Decision tree subtype classification results

Iteration	H Positions	N Positions
1	50, 278	54, 724
2	54, 674	61, 429
3	60, 156	69, 667
4	90, 651	80, 821
5	91, 282	81, 391
6	134, 353	82, 909
7	48, 408, 1080	115, 713
8	51, 343, 735	132, 538
9	53, 307, 622	145, 539
10	61, 127, 157	148, 637
11	89, 456, 959	151, 1674

12	93, 203, 657	179, 1737
13	103, 732,1766	190, 1648
14	104, 298, 495	238, 933
15	107, 207, 601	241, 1696
16	108, 473, 777	253, 1218
17	124, 450, 746	262, 799
18	141, 211, 927	323, 1649
19	150, 306, 459	335, 1518
20	153, 363, 759	343, 817
21	154, 483, 856	50, 372, 791
22	159, 268, 718	68, 385, 991
23	160, 280, 354	73, 426, 1407
24	162, 633, 994	77, 441, 1494
25	163, 223, 1088	78, 537, 1698
26	167, 476, 1089	196, 592, 1566
27	170, 687, 176	336, 1215, 1707
28	186, 680 1764	360, 1482, 1765

Table 4.3 Summary of H and N subtype informative positions at each of the iterations

4.4 Hidden Markov modeling of informative positions

To take advantage of informative positions, HMM profiles were built for each subtype and host for prediction purpose. HMM profiles provide statistical representation of a specific group of informative sites. These profiles were used in the Web prediction program to determine the subtype of a viral strain through sequence comparison. The hmmbuild was used to build the profile for a group of data. The -gapmax and –fast options were set to quickly and heuristically determine the architecture of the model. An algorithm referred to as BLOSOM was used to score the sequences.

The nucleotide base information corresponding to the informative positions generated by the decision trees were extracted from the training sequence data used for classification analysis. This collective information was used to build HMM profiles. Fig 4.4 shows the input format to the HMMER for building profiles. Each instance starts description with a maximum of 10 letters and the sequence data continues after a space. There should be no space within the description or the sequence data. Here the sequence data contains base information from only the informative positions. The hmmpfam option in the HMMER was used to search a single sequence against an HMM database.

```
Seq -GACAACAGTCCTTTTATCCATATTACAACTA-AAATGAAATCACATT-CAGAAGGATGAAA
Seq GGACAGCAATCCTTTTATCTATATTCCAACTA-AAATGAGACCACATT-CAGAAGGAAAAAA
Seq -GACAACAGTCCTTTTATCCATATTACAACTA-AAATGAAATCACATT-CAGAAGGATGAAA
Seq GGACAACAATCCTTTTATCCATATTACAACTA-AAATGAAACCACATT-CGGAAGGACGAAA
```

Fig 4.4 HMM input format

The HMM profiles were created from the input data hmmbuild option. Profiles were built separately for H1, H2, H3 and N1, N2. The individual H profiles and N profiles were concatenated together using cat option to form a single H profile and N profile respectively. Fig 4.5 shows a HMM profile for Human host. In the similar way profiles were constructed for each host (swine and human) and concatenated together for a segment. Profiles were built for HA, NA, NS, PA, PB1, PB2, and MP segments.

```
HMMER2.0
NAME  Human
LENG  78
ALPH  Nucleic
RF    no
CS    no
MAP   yes
COM   hmmbuild -F --fast --gapmax 1 --wblosum Human.hmm Human.fa
COM   hmmcalibrate Human.hmm
NSEQ  150
DATE  Tue Jun 16 16:01:10 2009
CKSUM 370
XT      -9967      -1   -1000   -1000   -9967      -1   -9967      -1
NULT       -1   -9967
NULE        0       0       0       0
EVD    -60.972248    0.230788
HMM          A       C       G       T
           m->m    m->i    m->d    i->m    i->i    d->m    d->d    b->m    m->e
           -582       *   -1591
      1     684    -326    -326    -326       1
      -       0       0       0       0
      -     -33   -6055   -7097    -894   -1115    -701   -1378    -582       *
      2    -326    -326     684    -326       2
      -       0       0       0       0
      -     -33   -6055   -7097    -894   -1115    -701   -1378       *       *
      .
      .
      .
     78       0       0       0       0      78
      -       *       *       *       *
      -       *       *       *       *       *       *       *       *       0
//
```

Fig 4.5 HMM profile for Human host

4.5 Subtype Web prediction system

In order to assist the task of detecting swine flu origin and serotype of a sequence, a web prediction tool for influenza A virus subtypes was developed. The prediction tool has been developed using LAMP technology. We have provided sample data and a simple tutorial on how to use the tool for host prediction. The tool allows users to submit sequences, choose the target serotype for subtype prediction, and select options to view the result [64].

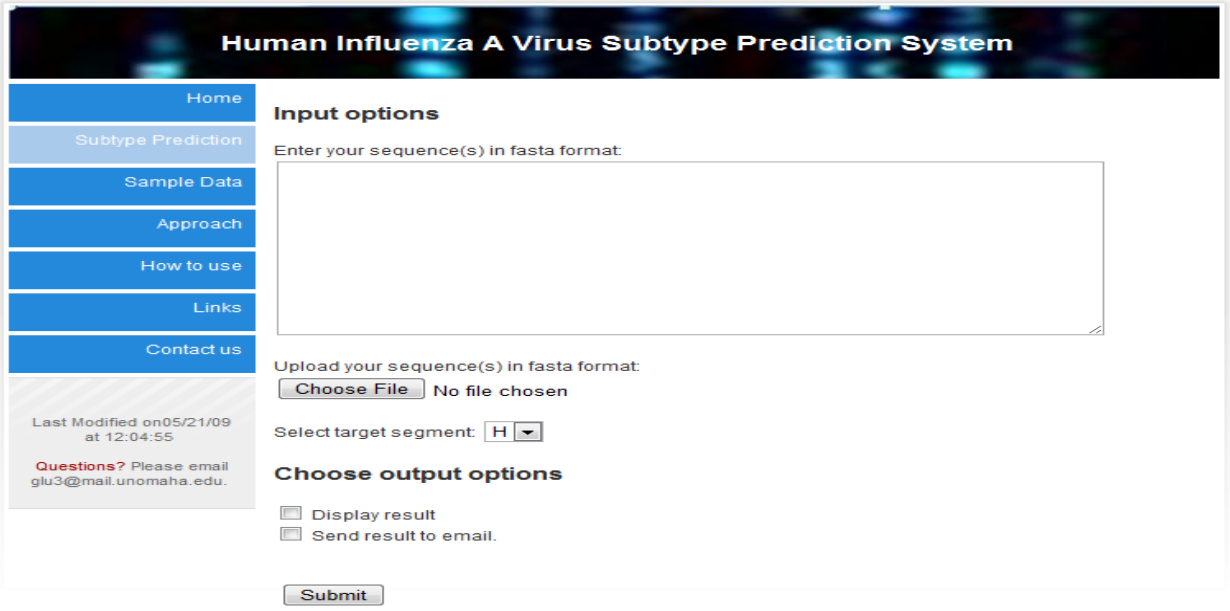

Fig 4.6 Influenza A virus subtype prediction system

The Website consists of a number of pages, including *Home*, *Predict Subtype*, *Sample Data*, *Algorithm*, *How to check*, *Links*, and *Contact Us* as shown in Fig 4.6. These pages provide the user an overview of the Web tool (*Home* page), sample sequences (*Sample Data* page) for testing web functions, a brief introduction of algorithms (*Algorithm* page), a simple tutorial (*How to check*), useful links (*Links*), and contact information (*Contact*). The major functioning page is the "*Predict Subtype*" page, which allows the user to submit sequences, choose appropriate options and view the result.

The given sequence or group of sequences are aligned and the informative positions are extracted to compare with the HMM profiles. The result of prediction will be displayed based on matching scores. An example is shown here on how to use the Web prediction system and what the result page looks like by analyzing a real sequence. A sample H1 sequence from NCBI is downloaded, submitted in text area, selected nucleotide type, checked selected options, and clicked Submit button. The prediction result is shown as in

Fig. 4.7, where the sequence is predicted correctly as H1 subtype, with a score 80.5 and an E value $3.3\times e^{-20}$. Score gives the probability of a given sequence to be related to the model and E-value measures the significance of profile match.

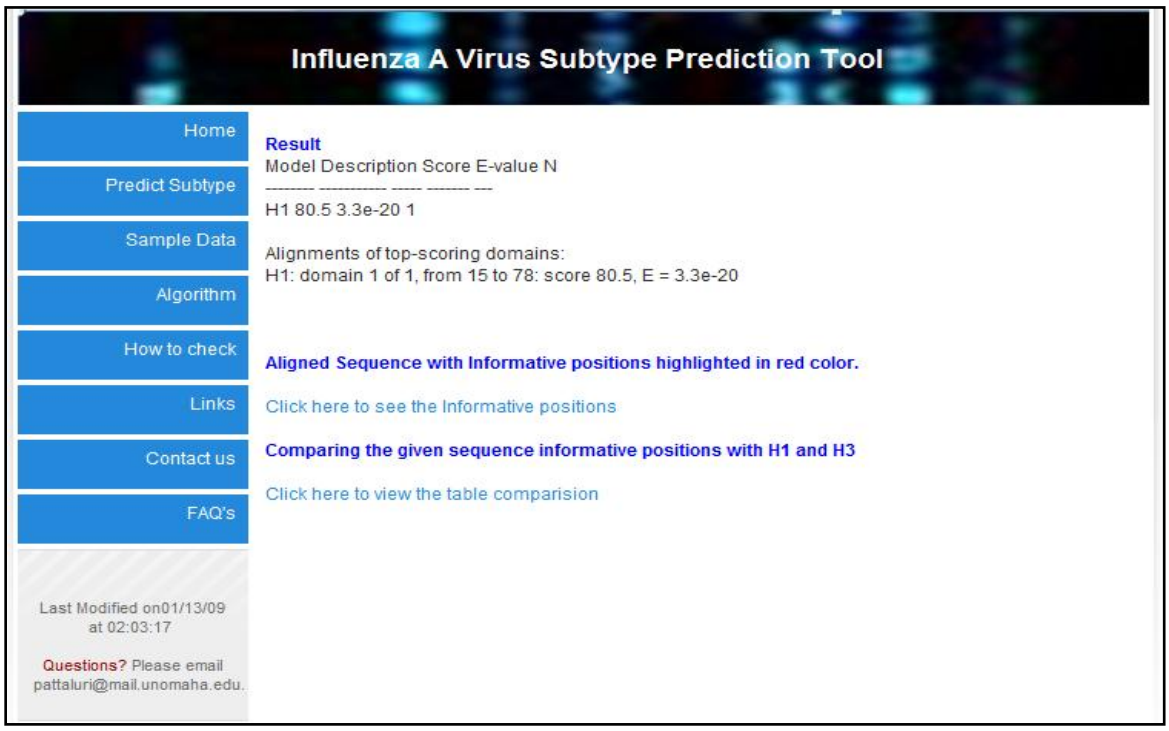

Fig. 4.7 The prediction result of a H1 sequence is shown, with two links for the user to view alignment and check informative positions, respectively.

Through performance testing, the system achieved an accuracy of 100% in predicting the subtype.

4.6 Mutation database

The informative positions found from the decision tree method were at nucleotide level. These positions were converted to amino acid positions based on their sequence and checked for the mutations. The positions differ from strain to strain and we checked for mutations for each strain and compared them with the mutation positions found through literature. The respective nucleotide and protein sequences for each strain were obtained

from initial data collected. The CDS information which represents the positions of conversion from nucleotide to amino acid was collected from the Entrez at NCBI [63]. The literature study was done by Ambreen Kedwaii [60], who reviewed papers from National Center for Biotechnology (NCBI), New England Journal of Medicine (NEJM) and other various bioinformatics journals to collect the mutation information.

The critical nucleotide positions identified through the decision tree system were 68 in total. For each strain the critical nucleotide positions were converted to the protein level and compared with the mutation positions found through literature. A total of 36 positions were found in common between the positions found through system and literature. A relational database was built in MySQL to store these mutations found through data mining using decision tree analysis and literature review.

4.7 Summary

This chapter demonstrates the use of integrating the decision tree and hidden Markov model methods. Three H subtypes: H1, H2, and H3; and two N subtypes: N1 and N2 are classified using decision tree analysis. A total of 78 most informative positions were found to collectively identify H subtypes; and 63 informative positions were identified for N subtypes. Using these positions HMM profiles were built for each subtype. A Web prediction system was built, which accurately detects the subtype of a given sequence by matching the sequence against the HMM profiles. The integrated approach of building HMM profiles based on the informative positions identified by the decision tree analysis achieved better prediction results. The informative positions are converted to protein level and compared with mutation positions found through literature study and a mutation database is built to store all these mutation positions.

Chapter 5

Applying machine learning techniques to classify H1N1 viral strains occurring in 2009 flu pandemic

The objective of this chapter is to describe the application of machine learning techniques for the classification of influenza hosts and detecting the origin of pandemic sequence data. Two classification analysis methods, decision tree and support vector machine, were employed to classify pandemic strains based on their host of origin. Phylogenetic analysis was also performed on a small set of data to compare the classification results. Profiles were developed using the informative positions found through decision tree analysis through hidden Marko modeling. A Web prediction system was developed to detect the origin of latest pandemic influenza virus using the HMM profiles.

Section 5.1 briefly describes the methodology. Section 5.2 provides the description of data collection and preprocessing. Section 5.3 reviews the phylogenetic approach for host classification. Section 5.4 presents the decision tree classification and HMM prediction results. Section 5.5 describes the implementation of support vector machine method for host classification.

5.1 Methodology

The methodology of classifying 2009 H1N1 viral strains using decision tree and support vector machine analysis is briefly described in the following steps. These steps are explained to detail in the following sections.

1. Collection and preprocessing the data. This includes collecting appropriate sequence data from online databases, and converting molecular sequence data to the required input forms for the analyses.
2. Phylogenetic analysis was applied to classify the sequences and describe the evolutionary relationships.
3. Decision tree analysis was carried out in the similar way as explained in the previous chapter to identify informative positions and model them as profile hidden Markov models to be used in the Web prediction system.
4. Support vector machine, an alignment free method, was applied for the classification analysis of influenza hosts.
5. Developing a Web prediction tool to assist the task of detecting swine flu origin.

5.2 Data collection and preprocessing

Three groups of influenza A H1N1 sequences were determined, including human (found only in human), swine (found only in swine), and human_swine (i.e., the latest pandemic influenza viral strains). Sequences were downloaded from the Global Initiative on Sharing All Influenza Data (GISAID) [65] and the National Center for Biotechnology Information (NCBI). Sequence comparison showed that the sequence dataset from GISAID is more complete compared with that of NCBI. Hence the sequences from GISAID were used in the analysis. When comparing sequences from different groups, we found that a number of sequences appeared in the human group also appeared in human swine group. These sequences were excluded for further analysis. Table 5.1 shows the count of sequences used for each segment and host. We restrict to a maximum of 150 sequences of each host and of each segment. Part of the data was used for training and the

remaining part was used for testing. The sequence data from segments HA, NA, PA, NS, PB1, PB2, M, and NP were used for SVM and decision tree analysis.

Segment	Human	Swine	Outbreak
HA	150	150	150
M	150	147	150
NA	150	148	150
NP	150	150	137
NS	150	147	111
PA	150	146	85
PB1	150	145	93
PB2	150	146	98

Table 5.1 Human influenza A virus sequences used for Host classification

The redundant sequences were eliminated from this data as explained in Section 4.1.2. Table 5.1 presents sequence count after eliminating redundancies. Nucleotide sequences were aligned using multiple sequence alignment as described in Section 4.1.3. Phylogenetic analysis was done using non aligned nucleotide data. Decision tree analysis was performed using aligned nucleotide data and support vector machine analysis was performed using protein data.

5.3 Phylogenetic analysis

Phylogenetic analysis is important to understand the evolution of species and of gene and protein families. The trees were generated using CLUTALW2.0 from EBI [66].

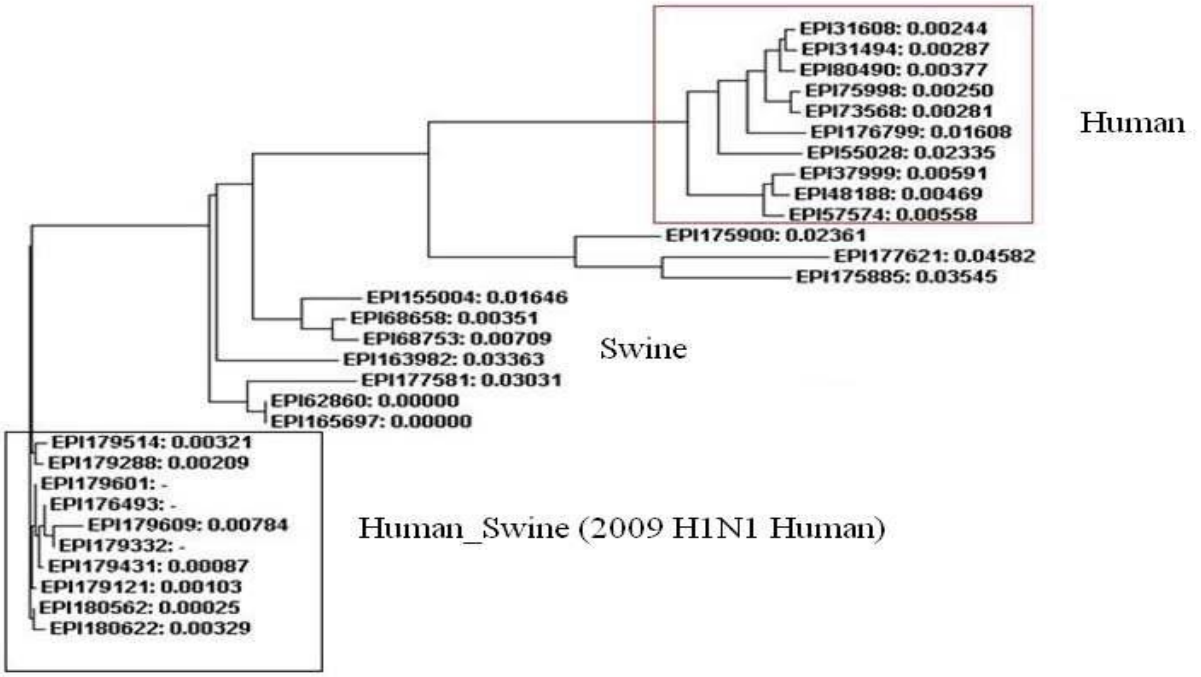

Fig. 5.1 Neighbor-Joining tree of influenza A viral strains based upon HA sequences

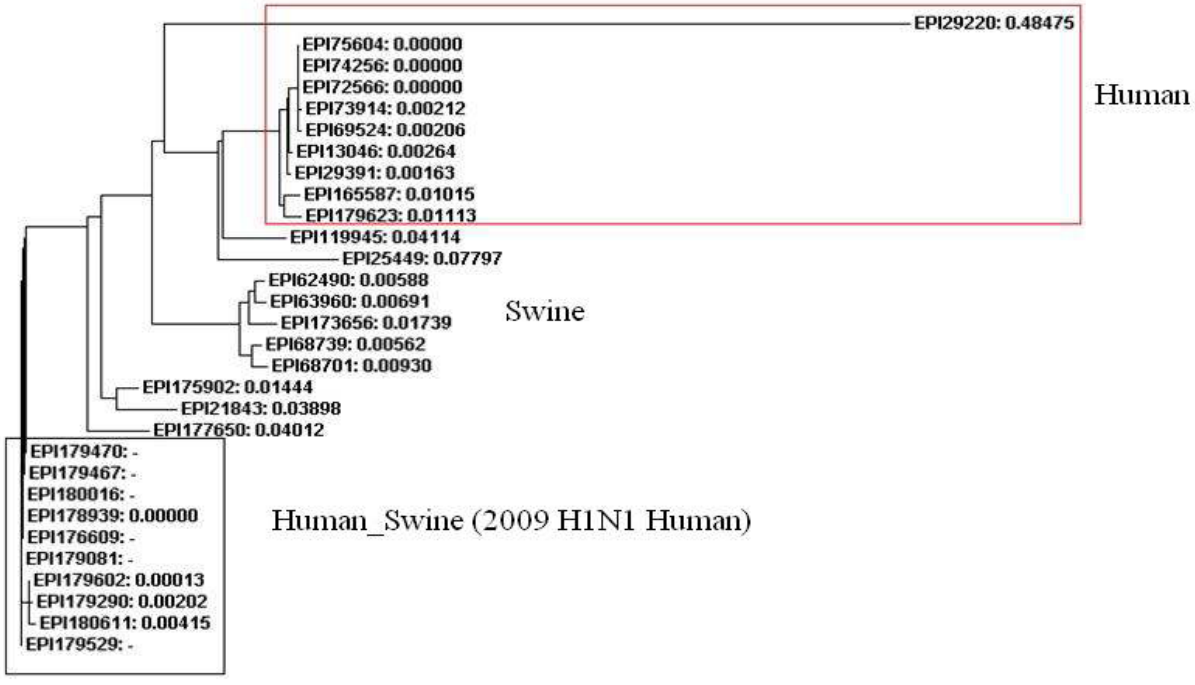

Fig. 5.2 Neighbor-Joining tree of influenza A viral strains based upon NA sequences

The tree diagrams (Fig 5.1 and 5.2) were made by using randomly selected 10 sequences from each group of human strain, swine strain and human_swine strains. Here, HA and NA segments are chosen as examples. The sequences in the top right corner of the diagrams (marked by a box) are from Human host. The sequences in the bottom left of the diagram (marked by another box) are from Human_Swine. The rest of the sequences are from Swine host. As presented in the tree diagrams, they indicate that the origin of 2009 H1N1 human influenza A is from swine rather than traditional human H1N1. In order to have a more comprehensive and confident result, alternative approaches using machine learning techniques were implemented.

5.4 Decision tree analysis for classifying influenza hosts

The decision tree method was implemented for the classification of hosts in the same way as explained in Section 4.1. The influenza host data was preprocessed first and J48 classifier from Weka was used for classification analysis.

5.4.1 Host classification results

A total of 25 iterations were performed for HA, NA, M, NP, PA, NS, PB1, PB2 segments. At each of the iteration steps, a decision tree was generated with one or more informative positions which classify two sets of data (i.e., Human, Swine). Table 5.2 summarizes the average accuracy of classification between Human, and Swine strains for each segment of all iterations. Sensitivity and specificity are statistical measures of the performance. Sensitivity measures the proportion of actual positives which are correctly identified and specificity measures the proportion of negatives which are correctly

identified. In this classification analysis, sensitivity measures the percentage of human sequences correctly identified as human host and specificity measures the percentage of swine sequences correctly identified as swine host.

Segment	Accuracy	Sensitivity	Specificity
HA	97.55%	96.18%	97.26%
M	96.38%	97.78%	97.08%
NA	98.12%	99.35%	98.24%
NP	98.28%	97.38%	98.16%
NS	97.22%	98.5%	98.78%
PA	98.13%	99.34%	99.12%
PB1	97.92%	96.88%	97.73%
PB2	96.86%	95.52%	97.84%

Table 5.2 Decision tree host classification results

Altogether 67 nucleotide positions for the HA segment and 64 nucleotide positions for the NA segment have been identified as informative by the decision tree analysis. The informative positions of HA and NA segments are summarized in Table 5.3. The sequence data from only these positions were collected together to form HMM profiles.

Iteration	HA Positions	NA Positions
1	27, 574, 671	47, 681, 728
2	28, 543, 408	48, 478, 421
3	29, 604, 625	124, 289, 837
4	331, 633, 634	281, 439, 793
5	404, 443, 500	267, 354, 721
6	597, 841, 1015	58, 293, 842
7	412, 725, 1036	654, 841, 994

8	590, 881, 944	176, 351, 1014
9	423, 623, 631	78, 112
10	37, 287, 448	731, 927, 1085
11	717, 857	493, 919
12	361, 580, 624	208, 643
13	867, 1111	585, 774, 1183
14	171, 873, 1126	89, 818, 1076
15	421, 556, 795	434, 1008
16	800, 846, 858	717, 982, 1126
17	32, 918, 1058	990, 1073
18	321, 446	256, 542,1045
19	162, 721	92, 548, 978
20	221, 230	144, 662
21	187, 1156	593, 834
22	254, 553	784, 929
23	282, 440	789, 822
24	726, 730, 771	419, 1031
25	316, 364, 473	680, 936

Table 5.3 Summary of HA & NA segments informative positions at each of the iterations

5.4.2 Profile modeling of informative positions

The informative positions identified from host classification data were modeled into profiles using hidden Markov modeling. The informative positions were processed into profiles as described in Section 4.4. The HMM profiles were created from the input data hmmbuild option. Profiles were built separately for swine and human. The individual

profiles were concatenated together using cat option to form a single host profile. Profiles were built similarly for HA, NA, NS, PA, PB1, PB2, and MP segments.

5.4.3 Host Web prediction system

A Web prediction tool was developed similar to subtype prediction system presented in Section 4.5. The given sequence or group of sequences are aligned and the informative positions are extracted to compare with the already existing HMM profiles. The result of prediction will be displayed based on matching scores.

5.4.4 Prediction of 2009 pandemic strains

To find the origin of the 2009 pandemic strains, outbreak sequences (human_swine) were collected and tested against the host Web prediction system. The results showed that 98% of the outbreak sequences are of swine origin. This suggests that latest outbreak sequences be more closely related to swine rather than human viral strains.

5.5 Support vector machine analysis for classifying influenza hosts

Support vector machine (SVM) is an alignment-free method which uses vectors to classify objects. The main advantage of SVM is that it does not depend on multiple alignment, thus it can avoid errors, if any, in multiple alignment files. SVM light tool is used for classification analysis [52]. Protein data was used for the support vector machine classification analysis.

In order to use SVM for classification, first the frequency was computed of each amino acid or a certain length of amino acids group. For example, sequence “GPPAV” can be treated in one letter a time (1-mer): “G” with frequency of 0.2 and “P” with frequency of 0.4, etc. Alternatively, it can be treated as two letters at a time (2-mer): “GP” with

frequency of 0.25 and "PP" with frequency of 0.25, etc. These amino acid frequencies were treated as vectors, and the distribution patterns of k-mer (up to 3) amino acids were used for classification analysis.

As standard SVM can only classify two classes of data, SVM analysis was implemented to classify human and swine hosts for HA, NA, M, NP, NS, PA, PB1, and PB2 segments. Classification results of all segments are provided in Table 5.4. In this classification analysis, sensitivity measures the percentage of human sequences correctly identified as human host and specificity measures the percentage of swine sequences correctly identified as swine host.

Segment	Accuracy	Sensitivity	Specificity
HA	96.32%	96.73%	95.1%
M	95.18%	94.33%	95.82%
NA	97.48%	95.67%	98.47%
NP	97.84%	96.3%	97.94%
NS	96.14%	95 %	97.52%
PA	96.66%	95.26%	96.82%
PB1	96.12%	97.33 %	96%
PB2	95.71%	94.67%	96.72%

Table 5.4 SVM host classification results

The latest outbreak sequences were tested against the support vector machine developed to classify human and swine hosts to predict their origin. For all the segments, the outbreak sequences were classified as swine origin.

5.6 Summary

Accurate detection of influenza viral origin can significantly improve influenza surveillance and vaccine development. Phylogenetic analysis of randomly selected sequences revealed significant differences between human and swine influenza A H1N1 viral strains. In this chapter, machine learning techniques were implemented to identify the evolutionary origin of the latest human pandemic influenza H1N1 viral strains. Both Support Vector Machine and decision tree methods agreed each other on that the viral strains causing the latest human pandemic are of swine origin. Along with the previous chapter, this study demonstrated the power of integrating the decision tree and hidden Markov model approaches in classifying influenza A viral subtypes and hosts.

Chapter 6

Applying neural networks to classify Influenza virus Antigenic types and hosts

This objective of this chapter is to provide detailed description of architecture and application of the artificial neural networks for the influenza analysis. A feed-forward backpropagation neural network is trained to predict important influenza virus antigenic subtypes (H1, H3 and H5) and hosts (Human and Swine). The chapter describes the architecture design including the number of layers, the number of processing units and interconnections between them, backpropagation training algorithm, network parameters used and preprocessing of data. The application of different input data encoding schemes is explained.

This chapter is divided into four sections. Section 6.1 overviews the methodology applied in the neural network analysis. Section 6.2 provides the architecture of the feed forward network designed. Section 6.3 describes preprocessing of nucleotide base data to binary representation. Section 6.4 describes the training of neural network through backpropagation algorithm. Section 6.5 and 6.6 presents the classification results of influenza subtype and host.

6.1 Methodology

The methodology of the application of backpropagation feed-forward neural network is briefly explained in the following steps.

1. Designing the neural network architecture. The considerations include selecting the network architecture (feed-forward network), training algorithm (backpropagation) and number of layers and units.
2. Preprocessing the data. This requires input sequence encoding methods to convert molecular sequences (character strings) into input vectors to the network. Several direct and indirect encoding schemes were applied.
3. Training the network. This includes selection and partition of training data (cross-validation), generalizing the network, selecting the network parameters and training the network with preprocessed data.
4. Post processing the outputs from the network to desired forms and transform them into classification results.

6.2 Feed-forward network architecture

A feed-forward neural network was applied for the classification analysis. The architecture of the network consists of three layers: one input, one hidden, and an output layer (Fig 6.1). A simple network with feed-forward structure: Signals flow from inputs, forwards through any hidden units, eventually reaching the output units. Networks of this type can be trained to provide a desired response to a given input. Learning through feed-forward networks is called supervised learning. The number of neurons in input, hidden, and output layers varies depending on the type of input given to the network and type of classification. The number of neurons in output layer directly depends on the number of classes to be classified. The network is fully connected. Each unit in input layer is connected to all units in the hidden layer which are further connected to every unit in the output layer.

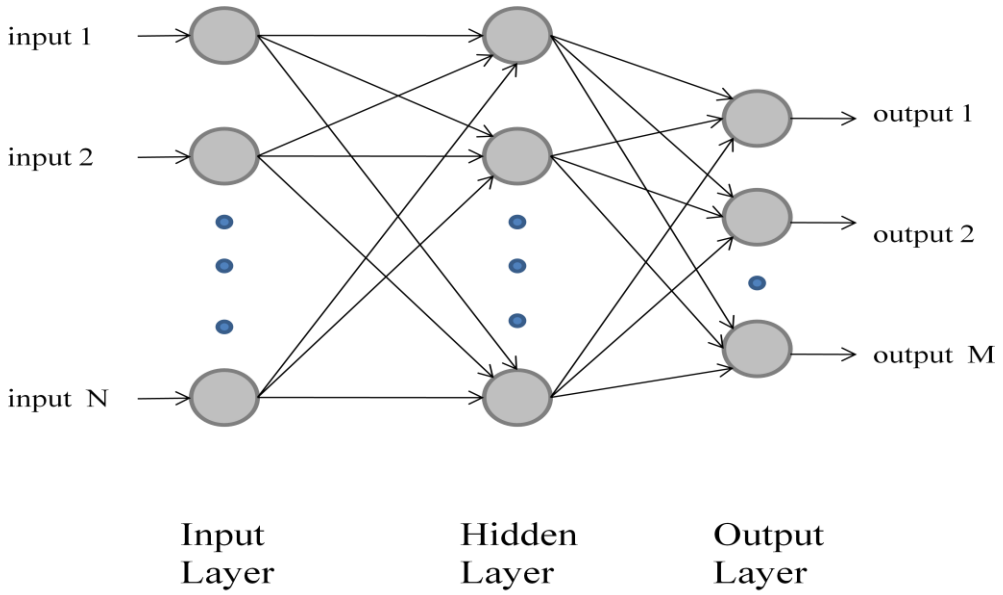

Fig 6.1 Architecture of neural network

The number of units in each layer for direct encoding is summarized in Table 6.1. The number of input units given to the network was directly related to number of bases in the sequence data. Two networks are used for the analysis. One network was given input from aligned data and hence contains gaps. Each instance in the training set consists of same base length and the number of input units is equal to that length. Another network was provided with non aligned data which does not contain gaps and different sequence instances have different lengths. Therefore, the length of the shortest sequence from the training data was selected as the number of input units. The indirect encoding scheme was applied with different set of frequencies. The number of units in the input layer was the number of frequencies generated by k-mer method, which is based on the value of *k* (Table 6.2). The number of units in input layer is calculated by n^k, where *n* is the number of bases (4 in this case) and *k* is the value of k-Mer frequencies. For example in 3-Mer encoding, where *k* is 3 and the number of units in the input layer is $4^3 = 64$.

There are several methods to select number of units in hidden layer [68]. A common strategy is the number of hidden units should be 2/3rd the size of input layer. Using this strategy as starting point, the hidden units were selected by trial and error basis. The number of output units was 3 in H subtype classification as it contains three classes (H1, H2 and H3) and 2 in all remaining cases.

	Input units		Hidden units		Output units
Segment	Type 1	Type 2	Type 1	Type2	
H	2125	1500	1600	1000	3
N	2063	1500	1600	1000	2
HA	1875	1700	1500	1200	2
M	1252	1200	900	900	2
NA	1588	1300	1200	950	2
NP	1874	1700	1250	1250	2
NS	923	800	600	600	2
PA	2135	2000	1600	1400	2
PB1	2178	2000	1600	1400	2
PB2	2216	2000	1650	1400	2

Table 6.1 Summary of direct encoding inputs

K-mer	Input units	Hidden units	Output units
2-mer	16	5	2
3-mer	64	42	2
5-mer	1024	600	2
10-mer	1048576	80000	2

Table 6.2 Summary of indirect encoding units

6.3 Data preprocessing

The key factor in training the neural network is the selection and pre-processing of the data. Pre-processing refers to the conversion of nucleotide data to be applicable as inputs to the network. Using informative and complete training data set plays a vital role in the success of the system. Data processing involves collection of data and pre-processing before feeding to the network. Nucleotide sequence data was used for training neural networks. The sequence data consists of letters A, C, G, and T; and the gap is represented as '-'. There are several encoding schemes to convert nucleotide bases in order to be presented as input [69].

An ideal encoding method should extract maximal information from the sequence data and satisfy the assumption that similar sequences are represented by close vectors. The selection of an appropriate encoding method is one of the significant factors that determine the performance of the system, because the encoding determines what information is presented to the network for the analysis. Hence two different approaches were used to encode sequence data, direct sequence encoding and indirect sequence encoding.

6.3.1 Direct sequence encoding

Direct sequence encoding uses binary numbers (0 and 1) to represent each nucleotide base. Two types of direct encoding schemes were used. In this encoding, the inputs to the network were given sequentially in the order of bases in the sequence data. They preserve the order of bases in sequence data.

The first approach was a dense encoding scheme, which uses only two units to represent four nucleotides (Type 1). Sequence data was not aligned in this case, hence no gap.

Base	Encoding
A	00
C	01
G	10
T	11

The second approach was a sparse encoding scheme which considers a vector of four units with three zeroes and a single one is used to represent four bases and gap is represented with all zeros (Type 2). The number of inputs in this case is equal to the length of the aligned sequence. The encoding scheme is presented below.

Base	Encoding
A	0001
C	0010
G	0100
T	1000
-	0000

The third approach is an alternative to the sparse encoding method uses five units to represent the nucleotide bases (Type 3).

Base	Encoding
A	00001
C	00010
G	00100
T	01000
-	10000

6.3.2 Indirect sequence encoding

Indirect sequence encoding includes calculating individual residue frequency or a group of residues as k-mer frequencies. The non aligned nucleotide sequence data was considered for calculating frequencies. The k-mer frequency method was tested with different values of k including 1, 2, 3, 5, 10, 20 etc. The count of input values differ based on the value of k. As described in section 6.2, the total number of frequencies calculated is given by n^k , where n is the number of bases i.e. 4 and k is the frequency used.

6.4 Backpropagation training

Backpropagation algorithm was implemented for training the network. The encoded input values are presented to the input layer units. The weights for input layer are assumed to be 1. No activation function was applied for the input layer. The raw values were presented to the next hidden layer. The inputs to hidden layer were computed using raw output from input layer, weight and bias values. The activation results of the hidden units based on the computed net input were then fed to output layer. The output of the network was reached by passing output of each unit as input to the following layer.

After each cycle, the resulting output from the network is compared with desired output and error approximation was calculated by using

$$E = O * (1 - O) * (D - O) \quad (1)$$

Where E is error value, O is output value computed from the network, and D is the desired output. In case of host classification, the desired output values are 1 for swine and 0 for human. The output values from the network are compared with the desired output values and the error value is computed.

Based on the error value, the weights and bias values were modified starting from the output layer to input layer.

$$\Delta b = \text{Learning rate} * E \quad (2)$$

$$\Delta w = \Delta b * I \quad (3)$$

Where Δb represent change in bias value, Δw_i represent change in weight value, and I denote the input value. Algorithm is summarized in following steps

1. Feed input to the network and compute every unit. This can be done by calculating net sum of all the inputs multiplied with respective weights. The net sum was presented through the activation function to generate the output from the unit.

2. Compute error function of the network by taking the sum of squared error of every unit in the output layer.

3. Compute error approximation of each unit in the output layer using formula (1). This can be done by using the calculated output and the desired output.

4. Calculate the error approximation of each unit in the hidden layer. The hidden error approximation depends on error values calculated for the output layer units.

5. Compute weight deltas using formula (3) and add them to each of the weights one layer at a time.

We have trained two networks for subtype classification. HA sequence data was supplied to one network to classify different H subtypes (H1, H2 and H3) and NA sequence data was supplied to another network to classify N subtypes (N1 and N2). We have trained one network for host classification. This network was trained to classify Human and

Swine sequences. The process of modifying weight and bias values was repeated until error approximation reaches an acceptable value (0.1). All the experiments achieved acceptable value (i.e. training algorithm was success) between 7000 – 10000 cycles. Hence the upper limit of the number of cycles was set to 10000. This is the number of cycles after which the output values from the network are close to the optimal. The step and sigmoid activation functions were used to scale the output of each unit in the network into proper ranges. Step function was used for direct encoding inputs and sigmoid function was used in the instance of indirect inputs.

6.4.1 Training parameters

There are several parameters set to run the network. The default weight was applied to all branches that connect units. Bias is simply a weight applied to an input whose value is always one or zero. Bias values do not affect the training algorithm. The default values were applied in the first cycle of the backpropagation training. After first cycle, weight and bias values were updated based on the error values. Modifying weight and bias values minimizes the error and improves the chances of the network to classify testing data accurately. Learning rate can be described as acceleration of the training algorithm. It affects the learning procedure speed of the network. Learning rate was also adjusted along with weight and bias values to speed up the training the network. Error approximation is the minimum difference accepted between the desired output and actual output from the network. Table 6.3 gives the description of parameters used in both direct and indirect.

Parameter	Direct values	Indirect values
Activation function	Step	Sigmoid
Default weight	0.1	0.05
Default bias	0.4	0.01
Error approximation	0.1	0.1
Learning rate	0.15	0.08

Table 6.3 Description of parameters

6.5 Subtype classification

Similar to decision tree classification described in section 4.3 and section 5.4, neural network analysis was applied to classify H and N subtypes. The subtype classification was performed for HA (Hemagglutinin) and NA (Neuraminidase). Among 16 HA antigens, three subtypes H1, H2, and H3 and among 9 NA antigens, two subtypes N1 and N2 are considered for classification analysis. Hence the number of neurons in the output layer are three for HA classification and two for NA classification.

The classification results using two direct encoding techniques are summarized in Table 6.4. For direct binary transforming of nucleotide sequences, Type 2 (using 4 bits to represent a nucleotide base) and Type 3 (using 5 bits to represent a nucleotide base), both encoding schemes produced comparable results in terms of accuracy, both reaching 92% accuracy. They performed better when compared to the use of a Type 1 encoding scheme (two bits to represent a base). The reason for a better performance by Type 2 and Type 3 encoding schemes is likely attributable to the inclusion of gaps.

Type	Accuracy	Sensitivity	Specificity
Type 1	91%	90%	91%
Type 2	92%	93%	91%
Type 3	92%	92%	93%

Table 6.4 Subtype direct encoding classification results

Table 6.5 presents the accuracy results of indirect encoding techniques. The accuracy for subtype classification is above 91% when the frequencies of k-mer nucleotide strings were used as input to the neural network. The use of 6-mers perform particularly well (98% accuracy) compared to the use of 2-mers (92%). The results show that a higher value of k achieves relatively better classification results. This result demonstrates that k-mer frequencies capture important features of nucleotide sequences for classification of viral subtypes. Overall performance of k-mer frequency methods was better than binary representation methods. This may be due to the fact that k-mer frequencies capture important features for determining the properties of nucleotide data.

k-mer	Accuracy	Sensitivity	Specificity
2-mer	92%	91%	93%
3-mer	95%	96%	92%
5-mer	97%	95%	98%
6-mer	98%	96%	99%

Table 6.5 Subtype indirect encoding classification results

6.6 Host classification

The classification analysis was performed for HA, NA, M, NP, PA, NS, PB1, PB2 segments. Two sets, Human and Swine, were classified using neural network. The performance results of all encoding techniques are presented in Table 6.6, 6.7, 6.8 and 6.9.

Segment	Human	Swine	Avian	Mean
HA	94%	90%	92%	92%
M	95%	85%	87%	89%
NA	95%	87%	91%	91%
NP	93%	88%	92%	91%
NS	93%	87%	87%	89%
PA	94%	89%	90%	91%
PB1	97%	90%	92%	93%
PB2	95%	88%	93%	92%

Table 6.6 Accuracy of Host classification (direct encoding, 4-bit)

Segment	Human	Swine	Avian	Mean
HA	96%	91%	95%	94%
M	93%	90%	90%	91%
NA	94%	92%	90%	92%
NP	94%	89%	90%	91%
NS	92%	85%	90%	89%
PA	96%	90%	93%	93%
PB1	97%	94%	94%	95%
PB2	95%	92%	92%	93%

Table 6.7 Accuracy of Host classification (direct encoding, 5-bit)

Segment	Human	Swine	Avian	Mean
HA	99%	96%	99%	98%
M	98%	96%	94%	96%
NA	100%	97%	97%	98%
NP	98%	96%	98%	97%
NS	98%	93%	95%	96%
PA	97%	96%	98%	97%
PB1	97%	92%	96%	95%
PB2	98%	95%	95%	96%

Table 6.8 Accuracy of Host classification (indirect encoding, 3-bit)

Segment	Human	Swine	Avian	Mean
HA	97%	92%	96%	95%
M	94%	91%	94%	93%
NA	97%	93%	95%	95%
NP	95%	94%	93%	94%
NS	98%	91%	96%	95%
PA	98%	95%	95%	96%
PB1	94%	91%	94%	93%
PB2	96%	92%	94%	94%

Table 6.9 Accuracy of Host classification (indirect encoding, 5-bit)

The latest outbreak sequence data was submitted to the network designed to classify Human and Swine classification. Table 6.10 presents the prediction results of 2009 pandemic outbreak sequences against the network. The percentages in the table indicate the percentages of latest outbreak sequences in the testing data sets that are classified as Swine strain. In other words, all sequences from HA, M, NA, P, NS, PA, PB1, and PB2

were classified as swine influenza, which means sequences in these segments are more closely related to swine origin.

Segment	Accuracy Direct	Accuracy Indirect
HA	94%	97.67%
M	94.67%	97%
NA	94.33%	97.33%
NP	92.67%	95.33%
NS	91.67%	95%
PA	94.33%	96.67%
PB1	93%	96.33%
PB2	93.67%	97.33%

Table 6.10 Outbreak prediction results

6.7 Summary

In this study, the feed-forward backpropagation neural network is applied for the classification analysis of influenza virus. A comprehensive experiment on different k-mers and different binary encoding types showed classification based upon frequencies of k-mer nucleotide strings performed better than transformed binary data of nucleotides. It has been found for the first time that the accuracy of virus classification varies from host to host and from gene segment to gene segment. In particular, compared to swine viruses, human influenza viruses can be classified with high accuracy, which indicates influenza virus strains might have become well adapted to their human host and hence less variation occurs in human viruses. In addition, the accuracy of host classification varies from genome segment to segment, achieving the highest values when using the HA and NA segments for human host classification.

Chapter 7

Conclusion

Machine learning techniques become standard alternative approaches for classification in Bioinformatics. These methods automatically learn from sequence data and find subtle differences in the classifications associated with real data. They are able to identify complex features like informative positions in the case of decision trees and classify different groups with high accuracy.

The aim of this thesis was to explore various machine learning techniques to classify and predict influenza virus subtypes and hosts. Accurate classification of influenza viral subtype and detection of virus origin can significantly improve influenza surveillance and vaccine development. In this study, decision tree, support vector machine, and neural network methods were applied for influenza subtype and host classifications. The research demonstrated the power of integration of decision tree and hidden markov model approaches for the prediction of influenza virus features. A web prediction tool was developed for detection of influenza origin and subtype.

7.1 Results

This section summarizes the work done and the main results obtained through the course of this research.

This thesis demonstrated the use of integrating the decision tree and hidden markov model methods and achieved better prediction accuracies compared to only decision tree prediction. A web prediction tool is developed which predicts the subtype and host of a

given sequence. We have used several data encoding schemes used to represent nucleotide sequences in the feedforward backpropagation neural network analysis and found that classification of influenza viruses using k-mer frequencies performs better than using transformed binary data of nucleotide bases.

7.1.1 Classification analysis

Decision tree, support vector machine, and neural network methods were implemented for the classification analysis. Decision tree and neural network were presented with nucleotide input data and support vector machine was presented with protein data. C4.5 algorithm was used to construct trees in decision tree analysis. Several informative positions were collected by generating multiple decision trees. Support vector machine analysis used frequencies of 3-mer group of amino acids. For neural network analysis, feedforward architecture was implemented with backpropagation training algorithm. Several input encoding schemes were tested and found that 5-mer frequency inputs achieved highest accuracy.

For subtype classification, H1-3 and N1-2 subtypes were classified for all segments. For host classification, human, avian, and swine origins were classified. The classification accuracies of all the methods were above 90%. The overall accuracy of all the three methods is presented in Table 6.1. The results show that the decision tree method is well suited classification of influenza data.

Method	Accuracy
Decision tree	96.5
Support vector machine	96.2
Neural network	95.1

Table 6.1 Overall classification accuracy

The overall accuracy was found by averaging the accuracies of both subtype classification and host classification. These results have shown that different groups of influenza are well distinguishable, which indicate there are significant differences between them. It has been found for the first time that the accuracy of virus classification varies from host to host and from gene segment to gene segment. The finding of the highest accuracy achieved using HA and NA for human host classification has significant implications in the development of a computational system for influenza detection and surveillance.

7.1.2 Web prediction tool

A prediction tool was developed to detect the subtype and host of an influenza sequence by modeling the informative positions identified by the decision tree. Hidden markov modeling technique was used to model the positions. The probabilistic modeling approach using the informative positions collectively for prediction achieved better results (95% accuracy) than prediction based on checking individual positions in the decision tree. The web tool has an overall accuracy of 97% in predicting subtype and host of a given new sequence and can be accessed at

http://glee.ist.unomaha.edu/~pattaluri/swine/predict.php

Latest outbreak sequences can be tested using the tool to identify the host of origin. This helps for early detection of influenza outbreaks and improves influenza surveillance.

7.1.3 Origin of 2009 outbreak

The latest outbreak sequences are tested using decision tree, support vector machine, and neural network methods to find the origin. All the methods agreed on that the viral strains causing the latest human pandemic are of swine origin. The outbreak sequences were collected from GISAID database and processed to use in the analysis. In the decision tree analysis, they were tested using the prediction tool to find the origin. Overall 98% of the sequences were predicted as swine origin. Support vector machine and neural network analysis also predicted that pandemic strains are of swine origin. This suggests that the outbreak sequences are more closely related to swine rather than human viral strains.

7.1.4 Mutation database

We have identified informative positions determined by the decision tree analysis for classifying different groups of data. These positions are at nucleotide level and are converted to amino acid positions based on their sequence and checked for the mutations. These positions differ from strain to strain and mutations are checked for each strain and compared them with the mutation positions found through literature. Among the 68 informative positions found through the decision tree, 36 were found in common with the positions found through literature. A mutation database is built with the positions found through data mining using decision tree analysis and literature review.

7.2 Future work

There are a number of areas that could be researched further to improve the classification analyses implemented.

7.2.1 Learning methods

The high accuracies of the techniques implemented may be due to overtraining. The training data can be collected from various different databases. The neural network classification was done using standard feed forward architecture and backpropagation training algorithm. It is worth investigating with different architectures and training algorithms. The Hidden markov model used for prediction involves enumerating all possible paths through the model and is slow in comparison to other methods such as the position specific score matrix (PSSM) method [70]. A position weight matrix assumes independence between positions in the pattern, and the scores are calculated at each position independently. Therefore, it should be faster compared with the model used in the study.

7.2.2 Influenza data

Influenza A virus consists of 16 H and 9 N subtypes and has various types of hosts, the prediction system functionality can be extended to be able to identify influenza virus sequences from different subtypes and hosts. With respect to further data mining, it would be interesting to examine more nucleotide informative sites found by decision tree analysis at the protein level and check whether such changes are correlated with protein function.

7.2.3 Web prediction tool

The current prediction system is built using decision tree and hidden Markov modeling. The Web prediction tool can be extended by using support vector machine and neural network classification system.

Appendix A

Data Format conversion tool: A genome sequence format converter

A.1. Introduction

There are different tools used in the field of bioinformatics for different purposes. Most bioinformatics tools require specialized input formats of their own for processing. Because of these requirements, it is often necessary to convert sequences from one format to another. Phylogenetic trees are important to understand the evolution of species and of gene and protein families. The phylogeny trees produced contain only short descriptions which cause redundancy in some cases. We have developed a format conversion tool, which converts sequences from one format to another and works for the problem of short description in phylogeny trees. We have developed two versions, web-based and standalone. Having a standalone tool is convenient for Bioinformaticians to convert large number of sequences from one format to another and does not require intervention for a long time. This Format Conversion Tool is developed in Java and PHP.

Availability: Both web-based and standalone versions of the format conversion tool are available at http://glee.ist.unomaha.edu/~pattaluri/conversion/index.php. The standalone version can be downloaded from Downloads section. The website also contains a tutorial and examples.

There are a wide variety of sequence formats and sequences can be read and written in any possible format. A sequence format is simply a text represented by ASCII codes. A sequence format is an arrangement of characters, symbols and keywords. Sequence

format define the original sequence, comments and description containing ID, location and time of origin etc. Basic functionality of a sequence format is to maintain the sequence data and information about the sequence.

There are almost twenty five to thirty sequence formats available. The reason for having so many formats is that sequence data is stored across many databases and each of them has their own format to store. For example, GenBank, DDBJ and EMBL etc. stores sequence data in their own format. Also most sequence analysis tools use their own format for the analysis. Some of these formats are similar to each other than others.

A sequence format contains unique identifiers to distinguish each sequence. Generally accession number and ID name are used as identifiers for most of the formats. Accession numbers are unique keys containing alphabets and numbers. No two sequences have the same accession number. Some databases share accession numbers, for example EMBL and GenBank have a common accession number for a sequences stored in their database. ID names provide some kind of information about the sequence. Normally, ID names are delimited representation of some characteristics about the sequence. For example, a sequence with ID name ACT_CAEEL represents Acetylcholine Caenorhabditis Elegans. In many cases, it is hard to find unique ID names and so most of the sequence formats have accession number or a mixture of both accession number and sequence ID. Most formats provide options to hold other information like keywords, references and comments etc.

Most sequence formats support multiple sequences in a single file but formats such as *gcg* and *plain* supports only a single sequence per file. Nucleotide sequences are

generally represented using IUBMB standard codes and protein sequences are represented using IUPAC standard one-letter codes.

Popular sequence formats include FASTA, Phylip, MSF, Clustal, Mega etc. The International Nucleotide Sequence Database Collaboration (INSDC) consists of DDBJ (Japan), GenBank (USA) and the EMBL Nucleotide Sequence Database (UK). The basic format underlying information in DDBJ, EMBL and GenBank is a flat file. The correspondence between all three individual file formats facilitates the exchange of data between each of these datasets. GenBank describes each sequence entry with literature references, functional data, the location of coding regions and the sequence. Similarly, EMBL and DDBJ also provide resources for biological and medical research data.

A.2. Sequence Format Conversion

Sequence format conversion can be used for converting sequence of one format to another for future analysis. We provided conversions for all major formats. We used FASTA format as a base for format conversion. In most cases, the given sequence format is first converted to FASTA format and then converted to the required format. This is because FASTA is simple, easy to manipulate and holds the information required for any format. In our conversion analysis, we followed the most general notation used for all formats. We have provided both sequential and interleaved options for Phylip and Mega formats.

General information about the sequence formats implemented are summarized in Table 1

Format	Extension	Gap	Description
FASTA	.fa, .fasta, .fas,	Yes	A sequence in FASTA format starts with a single-line description, followed by lines of sequence data. The description line is distinguished from the sequence data by a greater-than (">") symbol in the first column. It is recommended that all lines of text be shorter than 80 characters in length.
CLUSTAL	.aln	Yes	CLUSTAL alignment format represents the sequences aligned in blocks and is used to construct phylogeny trees. Clustal format files contain the word *clustal* at the beginning.
Mega	.mega	Yes	A data file containing mega sequences begins with *#MEGA* and *TITLE* keywords. Sequences can be interleaved and sequential. This format is primarily used in Molecular Evolutionary Genetics Analysis software.
Phylip	.ph, .phy	Yes	The basic Phylip format consist two numbers in the first line, the number of taxa, and the number of characters. The data starts from second line where the first ten characters of each line denote taxon name, followed by sequence data. Sequences can be interleaved or sequential.
Nexus	.nxs, .nex	Yes	NEXUS file format is comprised of blocks such as the taxa block, data block, sets block, etc. Each block starts with <*block name*> and ends with *end;*. Comments are included inside square brackets *[and]*.
EMBL	.emb	Yes	The EMBL flat format represents associated meta-information, feature coordinates, and annotations. A sequence entry starts with an identifier line *"ID"*, followed by further annotation lines. Original sequence starts with line *"SQ"* and the end of the sequence is marked by two slashes *"//"*.

GenBank	.gen, .gbk	Yes	A sequence in GenBank format starts with a line containing the word *LOCUS* and a number of annotation lines. A sequence starts with an identifier *"ORIGIN"* and the end of the sequence is identified by two slashes *"//"*.
NBRF	.nbrf	Yes	A sequence in NBRF format consists of three parts. First part is a line with a *'>'* symbol, followed by a two letter code, semicolon and the sequence identification code. Second part consists of a single line description. Third part contains the original sequence.
Table	.tbl	Yes	Table format contains two fields, sequence name and the original sequence. Two fields are separated by tab space. Sequence data is represented by upper case letters and no whitespace characters are allowed.
Plain	.plain	Yes	A plain sequence contains only the original sequence. It does not contain any description. Sequence in plain format contains only IUPAC characters.
Newick	.nwk	No	Newick format represents tree data in a text file. This is used to transfer trees between different programs. Newick format uses parentheses and commas to represent trees with edge lengths.
MultiFASTA	.fa, .fasta, .fas	Yes	Multi-Fasta format is a FASTA file containing multiple fasta sequences. Each sequence begins with *">"* symbol.

Table 1 A brief description about the formats implemented

We have provided several formatting options for the users to ease their future analysis. An option to remove gaps is provided, which removes the gaps inserted through the alignment and converts the original sequence to the required format. This works in most cases except formats like clustal, which require all the sequences should be of same

length and hence removing gaps do not produce correct results. Also selecting letter cases (upper or lower) is provided to format the converted sequences.

A.3. Phylogeny tree description coding

The phylogeny trees produced from tree analysis programs contain only short descriptions which cause redundancy in most cases. We solved the problem of redundancy and short descriptions. This is a two step process, coding and decoding.

The coding part takes a set of input sequences in fasta, clustal or mega formats and for each sequence the whole description is replaced with a short description of first ten characters. If there is any redundancy with the ten characters, then first 9 characters are considered and the last character is replaced with a unique number. This unique number is incremented each time a redundant sequence of this case is encountered. These short descriptions and original descriptions are stored in a table format. Then it replaces the original description in the user given sequences with the short descriptions. A match table with original and short descriptions and the converted sequences with short descriptions are given as output from this step. These converted sequences are easy to use in the tree analysis.

The decoding part takes the match table generated in the coding part and the tree data in newick format as input. It replaces the short description in tree data with the original description from match table. It gives the tree data with the original long description without any redundancies as output.

A.4. Sequence Formatting

Most of the tools do not allow sequence descriptions containing special characters such as '/', '|' etc. We have provided an option for replacing the special characters and white space with an underscore character '_' in the sequence description. Currently this option works for Nexus and FASTA sequence formats.

A.5. Converting FASTA to Weka ARFF format

Weka is a popular tool which contains a collection of machine learning algorithms for solving data mining problems. Attribute relation file format (ARFF) is the data format used by Weka. Format conversion tool allows users to convert their biological sequences into attribute relation file format. The users need to give nucleotide sequences in Fasta format and select the number of attributes.

A.6. Removing Duplicate Sequences

In large sequence files, it is often difficult to check for duplicate sequences and the tools may produce erroneous results because of this redundancy. An option is provided to remove the redundant sequences from the given data set for all the major sequence formats. The result will produce a non-redundant sequence file.

A.7. Applications

Format conversion tool is currently used for sequence analysis in Bioinformatics lab at University of Nebraska at Omaha. The web prediction system was developed using the LAMP strategy. LAMP stands for Linux, a popular operating system, Apache, the most commonly used web server, MySQL, and PHP. The conversion programs were developed in Java and the web integration was done using PHP. The standalone version

is provided as a jar file and runs on system installed with java runtime environment. The web tool consists of several pages. *Sample Data* page provides sample data of all formats to test the web functions. A simple tutorial (*Tutorial* page) is provided to show how to use the format conversion and coding of tree data.

References

1. Human genome project. (2009). ISCID Encyclopedia of Science and Philosophy. Retrieved November 22, 2009 from http://www.iscid.org/encyclopedia/Human_Genome_Projects
2. Udo Seiffert, Barbara Hammer, Samuel Kaski, and Thomas Villmann. Neural Networks and Machine Learning in Bioinformatics - Theory and Applications. *ESANN'2006 proceedings - European Symposium on Artificial Neural Networks Bruges (Belgium), ISBN 2-930307-06-4.*
3. Ethem Alpaydin."Introduction to Machine Learning". *The MIT Pres,* ISBN 0-262-01211-1.
4. Pat Langley. "Elements of machinge learning". *Morgan Kaufmann Publications*. ISBN 1-5586--301-8.
5. P. Larranaga, B. Calvo, R. Santana, C. Bielza. J. Galdiano, I. Inza, J. A. Lozano, R. Armananzas, G. Santafe, A. Perez and V. Robles. "Machine learning in bioinformatics, "*Briefings in Bioinformatics,* 7(1), 2006, pp. 86-112.
6. KawaokaY (editor). (2006). *Influenza Virology: Current Topics.* Caister Academic Press. ISBN 978-1-904455-06-6.
7. Hay, A; Gregory V, Douglas A, Lin Y. "The evolution of human influenza viruses. *Philos Trans R Soc Lond B Biol Sci* 356 (1416): 1861-70. doi: 10.1098/rstb.2001.0999. PMID 11779385.
8. G. Neumann, T. Noda, and Y. Kawaoka. "Emergence and pandemic potential of swine-origin H1N2 influenza virus," Nature, 2009, 459(7249), pp. 931-939.
9. World Health Organization, "Swine Influenza," April 2009, *http://www.who.int/csr/don/2009_04_27/ en/index.html.*
10. Garten et al. "Antigenic and genetic characteristics of swine-origin 2009 A(H1N1) influenza viruses circulating in humans", PMID: 19465683.
11. R. Moellering, Jr, "Avian influenza: the next pandemic?", Clinical Microbiology Newsletter, 28(13), pp. 97-101, (2006)

12. Tollis M, Di Trani L. Recent developments in avian influenza research: epidemiology and immunoprophylaxis. Vet J 2002;164;202-15

13. H. Chen , G. Deng, Z. Li, G. Tian, Y. Li, P. Jiao, L. Zhang, Z. Liu, R. G. Webster, K. Yu, "The evolution of H5N1 influenza viruses in ducks in southern China", *Proc Natl Acad Sci USA*, vol 101, July 2004, pp. 10452-7

14. T. F. Hatchette, D. Walker, C. Johnson, A. Baker, S. P. Pryor and R. G. Webster, "Influenza A viruses in feral Canadian ducks: extensive reassortment in nature," *J Gen Virol*, vol 85, Aug 2004, pp. 2327-37

15. S. F. Altschul, T. L. Madden, A. A. Schäffer, J. Zhang, Z. Zhang, W. Miller, D. J. Lipman, "Gapped BLAST and PSI-BLAST: a new generation of protein database search programs," *Nucleic Acids Res*, vol 25, Sept 1997, pp. 3389-402

16. Bouvier NM, Palese P (September 2008). "The biology of influenza viruses". *Vaccine* 26 Suppl 4: D49-53

17. Ghedin, E; Sengamalay N, Shumway M, Zaborsky J, Feldblyum T, Subbu V, Spiro D, Sitz J, Koo H, Bolotov P, Dernovoy D, Tatusova T, Bao Y, St George K, Taylor J, Lipman D, Fraser C, Taubenberger J, Salzberg S (October 20 2005). "Large-scale sequencing of human influenza reveals the dynamic nature of viral genome evolution". *Nature* **437** (7062): 1162–6

18. Carsten Kemena and Cedric Notredame. "Upcoming challenges for multiple sequence alignment methods in high-throughput era". Bioinformatics 2009 25(19):2455-2465

19. R. C. Edgar, "MUSCLE: multiple sequence alignment with high accuracy and high throughput," *Nucleic Acids Research,* vol 32, March 2004, pp. 1792-7.

20. M. A. Larkin, G. Blackshields, N.P. Brown, R. Chenna, P.A. McGettigan, H. McWilliam, F. Valentin, I.M. Wallace, A. Wilm, R. Lopez, J.D. Thompson, T.J. Gibson and D.G. Higgins, **"**Clustal W and Clustal X version 2.0," *Bioinformatics, vol* 23, Sept 2007, pp. 2947-8

21. J. D. Thompson. D.G. Higgins, and T. J. Gibson, "CLUSTAL W: improving the sensitivity of progressive multiple sequence alignment through sequence weighting,

positions-specific gap penalties and weight matrix choice," *Nucleic Acids Research,* vol 22, Nov 1994. pp. 4673-4680

22. C. Notredame, D. G. Higgins, and J. Heringa, "T-Coffee: A novel method for fast and accurate multiple sequence alignment," *J Mol Biol* vol 302, Sept 2000, pp. 205-17

23. N. Saitou, and M. Nei, "The neighbor-joining method; a new method for reconstructing phylogenetic trees", *Mol. Biol. Evol.* 1987, 4(4):406-25

24. Gewehr, J.E., M. Szugat, and R. Zimmer: BioWeka--extending the Weka framework for bioinformatics. Bioinformatics 23(5): p. 651-3 (2007)

25. Quinlan, J. R. 1986. Induction of Decision Trees. *Mach. Learn.* 1, 1 (Mar. 1986), 81-106.

26. Quinlan, J. R. C4.5: Programs for Machine Learning: Morgan Kaufmann Publishers, 1993.

27. J. R. Quinlan. Improved use of continuous attributes in c4.5. Journal of Artificial Intelligence Research, 4:77-90, 1996

28. Nello Cristianini and John Shawe-Taylor. *An Introduction to Support Vector Machines and other kerne-based learning methods.* Cambridge University Press, 2000. ISBN 0-521-78019-5.

29. Ingo Steinwart and Andreas Christmann. Support Vector Machines. Springer-Verlag, New York, 2008. ISBN 978-0-387-77241-7.

30. V. Vapnik. Statistical Learning Theory, Theoretical book. *Reference book on generalization, VC dimension, Structural Risk Minimization, SVMs.* ISBN : 0471030031.

31. Arbib, Michael A. (Ed.) (1995). *The Handbook of Brain Theory and Neural Networks.*

32. Raul Rojas. *Neural Networks A Systematic Introduction.* ISBN 3-540-60505-3, 1966.

33. Haykin, Simon (1988). *Neural Networks: A Comprehensive Foundation (2 ed.).* Prentice Hall. ISBN 0132733501.

34. Martin Anthony and Peter L. Bartlett. *Neural Network Learning: Theoretical Foundations.* Cambridge University Press, 1999. ISBN 0 521 57353.

35. Stuart Russell and Peter Norvig. *Artificial Intelligence A Modern Approach.* p. 578, 1969.

36. Terrence L. Fine. *Feedforward Neural Network Methodology.* Information Science and Statistics. ISBN: 978-0-387-98745-3, 1999

37. Arthur Earl Bryson, Yu-Chi Ho (1969). *Applied optimal control: optimization, estimation, and control*. Blaisdell Publishing Company or Xerox College Publishing. pp. 481

38. Gribskov, M., A.D. McLachlan, and D. Eisenberg: Profile analysis: detection of distantly related proteins. Proc Natl Acad Sci U S A 84(13): p. 4355-8 (1987)

39. S. R. Eddy, "Profile hidden Markov models", *Bioinformatic*, 14(9), 1998, pp. 755-63

40. Song, D. S., C. S. Lee, K. Jung, B. K. Kang, J. S. Oh, Y. D. Yoon, J. H. Lee, and B. K. Park. 2007. Isolation and phylogenetic analysis of H1N1 swine influenza virus isolated in Korea

41. Pederson JC. "Hemagglutination-inhibition test for avian influenza virus subtype identification and the detection and quantitation of serum antibodies to the avian influenza virus", *Methods Mol Biol. 2008; 436:53-66.* PMID 18370041.

42. Janice C. Pederson, "Neuraminidase-Inhibition Assay for the Identification of Influenza A Virus Neuraminidase Subtype or Neuraminidase Antibody Specificity", *Methods in Mol Biol*, *2008; 436:67-75*.

43. Fouchier, R.A., et al.: Characterization of a novel influenza A virus hemagglutinin subtype (H16) obtained from black-headed gulls. J Virol 79(5): p. 2814-22 (2005).

44. Hatchette, T.F., et al., Influenza A viruses in feral Canadian ducks: extensive reassortment in nature. J Gen Virol 85(Pt 8): p. 2327-37 (2004)

45. Viseshakul N. "The genome sequence analysis of H5N1 avian influenza A virus isolated from the outbreak among poultry populations in Thailand." Virology. 2004; 328(2):169-76

46. Altschul, S.F., et al.: Gapped BLAST and PSI-BLAST: a new generation of protein database search programs. Nucleic Acids Res 25(17): p. 3389-402 (1997)

47. Lu, G., et al., GenomeBlast: a web tool for small genome comparison. BMC Bioinformatics 7 Suppl 4: p. S18 (2006)

48. Umar Syed and Golan Yona. "Using a mixture of probabilistic decision trees for direct prediction of protein function", *Proceeding of the seventh annual international conference on Research in computational molecular biology, 2003, pg 289-300* ISBN: 1-58113-635-8.

49. Ashkan *Sami* and Makoto Takahashi "Decision Tree Construction for Genetic Applications based on Association Rulse, *IEEE TENCON 2005*, Melbourne, Australia, November 2005, pp.21-25.

50. Salzber SL, Delcher Al, Kasif S, White O. "Microbial gene identification using interpolated Markov models", *Nucleic Acids Rec,* 1980 Jan 15;26(2),544-8

51. Xiaojing Yuan, Xiaohui Yuan, Fan Yang, Jing Peng, Buckles, B.P.: Gene Expression Classification: Decision Trees vs. SVMs. FLAIRS Conference 2003: 92-97 (2003)

52. Scott E. Fahlman and Christian Lebiere. "The Cascade-Correlation Learning Architecture", in Advances in Neural Information Processing Systems 2, D. S. Touretzky (ed.), Morgan Kaufmann Publishers, Los Altos CA, pp. 524-532. 34.

53. R. *Reed*. "Pruning algorithms a survey". IEEE Trans. on *Neural Networks* 4 5 (1993), pp. 740–747.

54. Haykin, Simon. "Neural networks and learning machines", Pearson Education, Inc, ISBN-13: 978-0-13-147139-9.

55. Brunak S, Engelbrecht J, Knudsen S. "Neural network detects errors in the assignment of mRNA splice sites", Nucl. Acids Res. 18:4797-4801.

56. Borries Demeler and Guangwen Zhou. "Neural network optimization for E.coli promoter prediction", Nucleic Acids Research, 1991, Vol. 19, No. 7 1593-1599.

57. Wu, C. H., and Shivakumar, S. "Back-propagation and counter-propagation neural networks for phylogenetic classification of ribosomal RNA sequences." *Nucleic Acids Research* 22, 4291-4299.

58. Farber. R., Lapedes, A. and Sirotkin, K. "Determination of eukaryotic protein coding regions using neural networks and information theory." *Journal of Molecular Biology* 226, 471-479.

59. Dick de Ridder, Robert P. W. Duin, "Sammons Mapping Using Neural Networks: A Comparison", *Pattern Recognition Letters, Vol 18 Issues 11-13,* November 1997, Pages 1307-1316

60. Ambreen Kedwaii, Pavan Kumar Attaluri, and Guoqing Lu. "Establishing Influenza A viral mutation database through literature and data mining", *Biotechnology and Bioinformatics Symposium (BIOT-2009).*

61. Sigurdardottir, A.K., H. Jonsdottir, and R. Benediktsson: Outcomes of educational interventions in type 2 diabetes: WEKA data-mining analysis. Patient Educ Couns 67(1-2): p. 21-31(2007)

62. Frank, E., et al.: Data mining in bioinformatics using Weka. Bioinformatics 20(15): p. 2479-81 (2004)

63. Bao Y., P. Bolotov, D. Dernovoy, B. Kiryutin, L. Zaslavsky, T. Tatusova, J. Ostell, and D. Lipman. The Influenza Virus Resource at the National Center for Biotechnology Information. J. Virol. 2008 Jan;82(2):596-601.

64. Pavan K. Attaluri and Guoqing Lu. *Human Influenza A Virus Prediction System.* http://glee.ist.unomaha.edu/~pattaluri/index.php

65. P. Bogner, I. Capua, D. J. Lipman, and N. J. Cox. "A global initiative on sharing avian flu data" *Nature* 442, 981 (August 2006)

66. Larkin, M.A., et al.: Clustal W and Clustal X version 2.0. Bioinformatics 23(21): p. 2947-8 (2007)

67. Thorsten Joachims, *Making Large-Scale SVM Learning Practical.* LS8-Report, 24, Universitat Dortmund, LS VIII- Report, 1988.

68. Jeff, T Heaton. Introduction to Neural Networks for Java, Second Edition. *Heaton Research, Inc.*

69. Cathy H. Wu. Artificial neural networks for molecular sequence analysis. *Computers Chem.* Vol. 21, No. 4, pp. 237-256, 1997. PII: S0097-8485(96)00038-1.

70. Hae-Jin Hu, Phang C. Tai, Robert Harrison, Jieyue He, Yi Pan. Protein Secondary Structure Prediction Using Support Vector Machine With a PSSM Profile and an Advanced Tertiary Classifier