A Novel Multi-layered Epitopes and Genetic Sequence Network for Predicting Antigenic Phenotype of H1 Influenza A Virus in Swine

Anugrah Saxena

Follow this and additional works at: https://lib.dr.iastate.edu/creativecomponents

Part of the Artificial Intelligence and Robotics Commons, Bioinformatics Commons, Comparative and Laboratory Animal Medicine Commons, Computational Biology Commons, Computational Engineering Commons, Data Science Commons, Epidemiology Commons, Numerical Analysis and Scientific Computing Commons, Small or Companion Animal Medicine Commons, and the Software Engineering Commons

Recommended Citation
https://lib.dr.iastate.edu/creativecomponents/802

This Creative Component is brought to you for free and open access by the Iowa State University Capstones, Theses and Dissertations at Iowa State University Digital Repository. It has been accepted for inclusion in Creative Components by an authorized administrator of Iowa State University Digital Repository. For more information, please contact digirep@iastate.edu.
A Novel Multi-layered Epitopes and Genetic Sequence Network for Predicting Antigenic Phenotype of H1 Influenza A Virus in Swine

by

Anugrah Saxena

A thesis submitted to the graduate faculty in partial fulfillment of the requirements for the degree of

MASTER OF SCIENCE

Major: Computer Science

Program of Study Committee:
Dr. Oliver Eulenstein, Major Professor
Tavis Anderson
Phillip C Gauger

The student author, whose presentation of the scholarship herein was approved by the program of study committee, is solely responsible for the content of this creative component. The Graduate College will ensure this creative component is globally accessible and will not permit alterations after a degree is conferred.

Iowa State University
Ames, Iowa
2021

Copyright © Anugrah Saxena, 2021. All rights reserved.
DEDICATION

I would like to dedicate this creative component to my mother, my father and my partner without whose immense support I would not have been able to complete this work. I would also like to give special thanks to my uncle and brother who believe in my ability to succeed. I would also like to thank everyone who was there to guide and support in lows and highs.
# TABLE OF CONTENTS

<table>
<thead>
<tr>
<th>LIST OF FIGURES</th>
<th>iv</th>
</tr>
</thead>
<tbody>
<tr>
<td>NOMENCLATURE</td>
<td>v</td>
</tr>
<tr>
<td>ACKNOWLEDGMENTS</td>
<td>vi</td>
</tr>
<tr>
<td>ABSTRACT</td>
<td>vii</td>
</tr>
<tr>
<td>CHAPTER 1. OVERVIEW</td>
<td>1</td>
</tr>
<tr>
<td>1.1 Introduction</td>
<td>1</td>
</tr>
<tr>
<td>1.1.1 Epitope based classification of strains</td>
<td>2</td>
</tr>
<tr>
<td>1.1.2 Antigenic Drift</td>
<td>4</td>
</tr>
<tr>
<td>CHAPTER 2. METHODS AND PROCEDURES</td>
<td>5</td>
</tr>
<tr>
<td>2.1 Generate Multi-layered Network</td>
<td>5</td>
</tr>
<tr>
<td>2.1.1 Identify Epitope Sequences</td>
<td>7</td>
</tr>
<tr>
<td>2.1.2 Generate Distance Matrices</td>
<td>7</td>
</tr>
<tr>
<td>2.1.3 Convert distance matrices into graphs</td>
<td>8</td>
</tr>
<tr>
<td>2.1.4 Find communities within the graph layers</td>
<td>11</td>
</tr>
<tr>
<td>2.2 Analyze User Input Sequences</td>
<td>12</td>
</tr>
<tr>
<td>2.2.1 Append user sequences into the multi-layered network</td>
<td>14</td>
</tr>
<tr>
<td>2.2.2 Identify the communities for sequences</td>
<td>15</td>
</tr>
<tr>
<td>2.2.3 Generate strain ranking for the sequences</td>
<td>15</td>
</tr>
<tr>
<td>CHAPTER 3. RESULTS</td>
<td>16</td>
</tr>
<tr>
<td>3.1 Introduction</td>
<td>16</td>
</tr>
<tr>
<td>3.1.1 Epitope based classification of strains</td>
<td>16</td>
</tr>
<tr>
<td>3.1.2 Antigenic Drift in US Gamma Clade (1A.3.3.3)</td>
<td>16</td>
</tr>
<tr>
<td>3.1.3 Analyzing user sequence against gamma strains based network</td>
<td>19</td>
</tr>
<tr>
<td>CHAPTER 4. SUMMARY AND DISCUSSION</td>
<td>25</td>
</tr>
<tr>
<td>4.1 Introduction</td>
<td>25</td>
</tr>
<tr>
<td>4.1.1 Epitope result different from sequence</td>
<td>25</td>
</tr>
<tr>
<td>4.1.2 Antigenic Drift through gamma clade study</td>
<td>25</td>
</tr>
<tr>
<td>4.2 Future Work</td>
<td>26</td>
</tr>
<tr>
<td>BIBLIOGRAPHY</td>
<td>27</td>
</tr>
</tbody>
</table>
**LIST OF FIGURES**

<table>
<thead>
<tr>
<th>Figure</th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>Figure 1.1</td>
<td>H1 HA Epitopes</td>
<td>2</td>
</tr>
<tr>
<td>Figure 1.2</td>
<td>Receptor binding sites</td>
<td>3</td>
</tr>
<tr>
<td>Figure 2.1</td>
<td>Multi-layered Network Architecture</td>
<td>6</td>
</tr>
<tr>
<td>Figure 2.2</td>
<td>Cb Epitope H1 HA edge weight distribution including same sequence data</td>
<td>9</td>
</tr>
<tr>
<td>Figure 2.3</td>
<td>Cb Epitope H1 HA edge weight distribution excluding same sequence data</td>
<td>10</td>
</tr>
<tr>
<td>Figure 2.4</td>
<td>Sb Epitope H1 HA egonet graph before edge removal</td>
<td>11</td>
</tr>
<tr>
<td>Figure 2.5</td>
<td>Sb Epitope H1 HA egonet graph after edge removal</td>
<td>12</td>
</tr>
<tr>
<td>Figure 2.6</td>
<td>Sb Epitope H1 HA egonet graph after edge removal with label</td>
<td>13</td>
</tr>
<tr>
<td>Figure 2.7</td>
<td>Sb Epitope H1 HA community graph with three communities</td>
<td>14</td>
</tr>
<tr>
<td>Figure 3.1</td>
<td>H1 HA Sb Epitope with user sequences</td>
<td>17</td>
</tr>
<tr>
<td>Figure 3.2</td>
<td>H1 HA Strain Ranking</td>
<td>18</td>
</tr>
<tr>
<td>Figure 3.3</td>
<td>Sa Epitope Gamma</td>
<td>19</td>
</tr>
<tr>
<td>Figure 3.4</td>
<td>Sb Epitope Gamma</td>
<td>20</td>
</tr>
<tr>
<td>Figure 3.5</td>
<td>Ca1 Epitope Gamma</td>
<td>20</td>
</tr>
<tr>
<td>Figure 3.6</td>
<td>Ca2 Epitope Gamma</td>
<td>21</td>
</tr>
<tr>
<td>Figure 3.7</td>
<td>Cb Epitope Gamma</td>
<td>21</td>
</tr>
<tr>
<td>Figure 3.8</td>
<td>Strain Epitope Gamma</td>
<td>22</td>
</tr>
<tr>
<td>Figure 3.9</td>
<td>Gamma Result</td>
<td>23</td>
</tr>
<tr>
<td>Figure 3.10</td>
<td>Gamma Network with Non-Gamma sequences</td>
<td>24</td>
</tr>
</tbody>
</table>
NOMENCLATURE

**Biological Terms**

- **IAV**  Influenza A Virus
- **HA**  Hemagglutinin
- **NA**  Neuraminidase

**Computational Terms**

- **QL**  Quartile
ACKNOWLEDGMENTS

I would like to thank my committee members: Dr. Oliver Eulenstein, Dr. Tavis Anderson and Dr. Phillip C Gauger. I would like to take this opportunity to express my thanks to those who helped me with various aspects of conducting research and the writing of this thesis. First and foremost, my major professor Dr. Oliver Eulenstein for his guidance, patience and support throughout this research. He always makes extra time to listen to any question and give advice for the problem. I would like to thank Dr. Phillip C Gauger for funding the research, his words of encouragement have often inspired me and renewed my hopes for completing my graduate education. I would also like to thank Dr. Tavis Anderson for his insight in Computational Biology and to look into the implications far into the future. I would like to give my special thanks to I would additionally like to thank Dr. Michael Zeller and Dr. Aditi Sharma for the research discussions and help with this creative component report. I would like to thank Dr. Alexey Markin for his guidance throughout the initial stages of the research and for his inspirational talks.

This research used resources provided by the SCINet project of the USDA Agricultural Research Service, ARS project number 0500-00093-001-00-D.
ABSTRACT

Influenza A viruses (IAV) in swine constitute a major economic burden to an important global agricultural sector, impacts food security, and is a public health threat. Despite significant improvement in surveillance for IAV in swine over the past 10 years, sequence data have not been integrated into a systematic vaccine strain selection process for predicting antigenic phenotype and identifying determinants of antigenic drift. We propose a novel pipeline that incorporates both the genetic sequence and the antigenic data of the hemagglutinin protein, identifies immune epitopes within the protein, and creates a multi-layered network that allows inference of antigenic drift. This method can be applied to identify new IAV strains that have distinct epitopes, and determine which IAV strains are central within the multi-layer network that explain observed diversity, and aid in the selection of robust candidates for multi-valent IAV vaccines. Our method on a smaller test dataset was able to give us an interesting cluster formation between different graph layers that can help understand results from different epitopes. Also, by increasing dataset sizes, we noticed the evolution of the H1 HA US gamma clade into multiple sub clades than can be used to programatically interpret antigenic drift.
CHAPTER 1. OVERVIEW

Influenza A virus (IAV) in swine causes economic burden to the agriculture, food, and swine production industries Holtkamp et al. (2013). Swine IAV also poses zoonotic threat to public health Haden et al. (2012), Holtkamp et al. (2007). The first recorded and verified human outbreak was the 1918 Spanish flu H1N1 that resulted in 500 million positive cases and 50 million deaths worldwide during the period of 1917-19 Koen (1919), Morens and Fauci (2007). IAV has persisted in humans for over a century. IAV consists of several genes (8) among which HA and NA have the highest rate of mutations Saitou and Nei (1986). Due to inter species transmissions, the swine population gets infected with Influenza A virus from humans in close proximity and aquatic birds migrations Webster et al. (1995). Swine IAV was first recognized in 1918 around the same time when Spanish Flu pandemic was circulating Koen (1919). Swine IAV was first isolated in 1930 Shope (1931). Since the start of the swine surveillance program in the United States, the gamma clade has been the most frequently detected H1 in swine Zeller et al. (2018). Swine IAV has more genetic diversity compared to the human IAV, making it difficult to develop effective, broadly protective vaccines. Developing faster and more reliable ways to identify antigenic drift of IAV can reduce the cost impact for industries, farms, veterinarians and human population in general.

1.1 Introduction

At present, multiple methods are being used to find genetically similar sequences. BLAST identifies similar sequences based on local alignment of sequences to find the closest matching sequences. Another method includes development of a phylogeny to see how does a sequence align in the phylogenetic tree with other lineages. The FLUture tool developed at ISU provides uses logistic regression to classify a sequence based on past sequence classification Zeller et al. (2018).
These methods though effective only rely on the sequence data as a whole.

We propose to use epitope site data along with the sequence data to do this classification.

Epitopes are the antibody binding surfaces of an antigen. They can be represented as short amino acid sequences. Prior determined epitope sites for H1 HA are Sa, Sb, Ca1, Ca2, Cb \text{Catton et al.} (1982), see Figure 1.1. Given this property, epitopes are one major factor that needs to be tested to identify antigenically similar strains.

For this project, we have the following hypotheses that we want to test.

1.1.1 Epitope based classification of strains

Can we find strains similar to a particular sequence of interest by comparing epitope data?
Figure 1.2: Receptor binding sites
1.1.1.1 Vaccine development

Is it possible to identify some strains using epitope data that can help us better understand along with be a candidate for vaccine development.

1.1.2 Antigenic Drift

Is it possible to detect antigenic drift in H1 HA subtype clades and can this be achieved by using epitope data for strains?

In the following chapters, we will discuss the workflow architecture that we developed using epitope data. Next, we will discuss the result for our experiments and summarize our findings.
CHAPTER 2. METHODS AND PROCEDURES

In this project, a pipeline was developed to assist users analyze network made from epitope and sequence data and test their strains of interest with it to identify patterns in the network and ranked strains similar to their own. This pipeline was divided into two sections. The first section of this pipeline is the network layer on which the multi-layered network will be generated. This layer is updated periodically when new strains are added to the network. The network layer’s state along with other significant parameters are stored for the user layer’s computation. The second section of this pipeline is the user layer. On the user layer, sequences to be analyzed are submitted, and subsequently appended to the multi-layered network. Communities encapsulating the sequences are identified and ranking of similar strains are provided to the user. The community graphs and ranked strains corresponding to the input sequences helps user to computationally observe the underlying importance of epitopes for the identification of strains to be a potential vaccine component.

The whole process was achieved by creating three different Jupyter Notebook files and an antigen class file. The workflow is illustrated in Figure 2.1. Each of the following sections further expand on the computational aspect of analyzing the IAV strains.

2.1 Generate Multi-layered Network

To generate the multilayered network, we started by identifying the epitopes within the HA gene based on prior literature. After extracting these epitopes from sequences derived from ISU VDL’s FLUture dataset, distance matrices based on Levenshtein distance method were computed. Subsequently social network graphs were generated using NetworkX graph library employing the distance matrices. This section will include identifying epitopes, generating distance matrices, generating social network graphs from the distance matrices and reducing the edges by
Figure 2.1: Multi-layered Network Architecture
determining cutoff threshold for sequence distances, identifying communities within the different layers of the network.

2.1.1 Identify Epitope Sequences

Using the object-oriented design in python, an antigen class was created inside antigens.py file. The class consists of H1 Antigenic Sites - H1 numbering (signal peptide removed) developed by Caton et al. (1982) and BLOSUM62 substitution matrix developed by Henikoff and Henikoff (1992) to take into consideration parity between amino acids and use it as a measure for assigning distance between sequences.

Levenshtein distance between sequences is calculated using the python-Levenshtein method, though direct computation of [global/local] sequence similarity through dynamic programming has been considered for future releases. The benefit of using dynamic programming is that instead of zero/unit distance, they will be replaced by the gap, match and mismatch scores.

The convertFastaToDictionary method takes in a sequence FASTA file as input and generates a dictionary of sequences. The findEpitopesFromSequences method takes in the sequence dictionary and for each sequence finds all the amino acids at corresponding epitope sites for each of the antigens (Sa, Sb, Ca1, Ca2, Cb) and combines them respectively to form epitope sequences.

2.1.2 Generate Distance Matrices

The DistanceMatrixGenerator notebook runs periodically and is supplied a nucleotide sequence FASTA file. It creates an object of antigen class as defined in section 2.1.1. For each strain in the FASTA file, a dictionary object is created for each epitope using the previously mentioned findEpitopesFromSequences method.

Distance matrices are then created for all the layers by finding Levenshtein distance between sequence pairs. This algorithm was optimized by solving for the lower half of the symmetric matrix and transposing the values. The run time to create the distance matrix is

\[ O(n^2L) \]
where $n$ is the number of strains used for the network, $L=O(nm)$ is time complexity to find Levenshtein distance between two strings of length $n$ and $m$. Save the raw distance matrices by pickling on the network side.

### 2.1.3 Convert distance matrices into graphs

Before diving further into creating a graph using distance matrices we need to define the complete picture in the form of a multi-layered network where each layer consists of graph and the layers connect via original sequences vertically.

#### 2.1.3.1 Defining the multi-layered network

To realize a distance matrix into a graph $G(V, E)$, we would need to first define the graph. Our multi-layered network $N$ can be defined along with it as well,

$$ N = \sum_{i=1}^{i=6} G_i, $$

where $G_i$ represents a graph for each epitope and original strain $G_i(V, E)$,

where, $V =$ sequences corresponding to epitopes/strain

and $E =$ edges with weight corresponding to the adjacency matrix vertices score $L(V_1, V_2)$

$G_i$ across the network are connected through the original strain allowing 3D traversal.

Defining the graph allows context for application of graph algorithms to identify communities that exist. The resultant graph is highly connected, increasing the computational complexity and time required to identify distinct communities. Further methods were employed to prune the graph and create sparsity.

#### 2.1.3.2 Make the graph sparser

The edges in graph $G_i$ can mean that the sequences in question are either same, very similar or very dissimilar. We want to identify and remove the edges that are between very dissimilar
After generating and analyzing the distributions for incrementally increasing networks (H1 HA and Gamma network strains separately) it was found that the second quartile values for distances were the closest for all the layers distributions to qualify as cutVal. It was noticed that the vertex pairs with same sequences were the ones with very high frequency, see Figure 2.2, thus disrupting the quartile values at times. In some situations, the first quartile value was coming as zero. This would mean that if programmatically quartile 1 is used for cutoff than any other pair with close similarity would have been disregarded causing in major loss of information. In order to overcome this issue, same sequence scores were discarded from the distribution, see Figure 2.3. Quartile 3 and quartile 4 distances were observed to be relatively low compared to the inter-quartile range (IQR), thus quartile 2 was selected as the preferred cutVal edge elimination threshold.

Figure 2.2: Cb Epitope H1 HA edge weight distribution including same sequence data
Figure 2.3: Cb Epitope H1 HA edge weight distribution excluding same sequence data
Figure 2.4: Sb Epitope H1 HA egonet graph before edge removal

Python’s *NetworkX* package was used to create the graphs by first converting the *numpy* distance matrix into graph object. Then, the node with the largest degree was identified. *Egonet* of the largest hub in the graph was generated by converting NetworkX graph object into an instance of *ego_graph*. *Spring_layout* was used to draw the graphs, see Figures 2.4 and 2.5, and *relabel_nodes* was used with strain define information to label them, see Figure 2.6.

### 2.1.4 Find communities within the graph layers

*Girvan and Newman (2002)* proposed a graph partitioning to detect community structures in social and biological structures. The property of these communities was that inside the communities the network nodes are tightly connected whereas the betweenness among communities is weak. The Girvan-Newman algorithm iteratively removes edges from the graph by default with the edge betweenness centrality value. After each iteration, the edge betweenness centrality value is calculated and the strength of the community determined *Newman and Girvan (2004)*. This produces desired densely connected nodes across different layers on the network. *Girvan – Newman* algorithm was applied on the labelled *NetworkX* graph object to get the
Since, weaker edges were already removed in section 2.1.3.2, thus only a few iterations of this algorithm would give us the desired communities. The graphs along with cutVal; and strain information thus generated in the network layer were pickled. This information will only be updated periodically when some new strains are available to be included in building up the network.

### 2.2 Analyze User Input Sequences

This section is exposed for the users in the form of a Jupyter notebook. The users can submit their sequences of interest for analysis with respect to the network generated in the previous section and make inferences from the strain rankings and graph communities.
Figure 2.6: Sb Epitope H1 HA egonet graph after edge removal with label
2.2.1 Append user sequences into the multi-layered network

The SequencesIdentityTool notebook gets sequences of interest as input along with the unpickled strain network data. Follow steps similar to section 2.1.1 to identify epitope sequences within the new sequence data. Following steps are taken to generate updated distance matrices including the new sequences along with the strains in the network.

2.2.1.1 Generate updated distance matrices

With the incoming sequences to be analyzed the whole network will be updated. Let there be originally $n$ strains that were used to generate the network and let there be $k$ new sequences. The new graphs on the network $N'$ will be as follows:

$$N' = \sum_{i=1}^{i=6} G'_i,$$

$$G'_i = G(V', E'),$$ where

$$V' = V + k,$$ and

$$E' = appended\ adjacency\ matrix\ such\ that$$

$$adj(G') = adj(G) + zeros(V + k, V + k)$$
For the uninitialized values across all the layers of the network compute similar to the process mentioned in section 2.1.1 and section 2.1.2 except this time it needs to be done only \( k \) times. Thus, this portion of computation would take \( \frac{k(n+k)}{n} \) fraction of the original time to retrieve the appended distance matrix.

### 2.2.2 Identify the communities for sequences

Steps to convert the distance matrices into graphs as in section 2.1.3 were repeated. Saved \( \text{cutVal}_i \) values were used to remove weak edges from the layers. Communities were identified using the Girvan-Newman algorithm as was done in section 2.1.4.

### 2.2.3 Generate strain ranking for the sequences

After gathering the list of strains within each community across each of the network layers, generate similar strain ranking was generated using naive approach. The naive ranking method gives a +1 for each occurrence of the strain within a community along with the sequence across network’s layers. The sorted top desired strains are denoted as the closest strains for the sequence of interest in question.
CHAPTER 3. RESULTS

Though the ranked results generated by the workflow are of importance, but they are not the only or the most important inference we can make.

3.1 Introduction

The following analysis was done on the mem node of the USDA Ceres server. Different sized datasets of H1 HA and H1 HA gamma clade strains with increasing sizes were used to generate the results. Though the following dataset consists of randomly chosen 51 H1 HA strains and 40 gamma strains only.

3.1.1 Epitope based classification of strains

Figure 3.1 makes it obvious that the gamma’s are clubbing to form its own community structure. Rest of the strains pertaining to different clades were bunched separately or may be with some gamma’s. It can also be noted that within the interconnected nodes of egonet graph several nodes of interest can be identified that connect with most of the graph nodes.

Looking at the ranking generated, see Figure 3.2, through process defined in section 2.2.3 shares the insight that the alpha sequence in question has several alpha’s, delta’s and gamma strains similar to it through the community based naive ranking. This though gives evidence that the ranking function does not do justice to other inference that can be made through analysis of graphs alone. We will expand on this further in Chapter 4.

3.1.2 Antigenic Drift in US Gamma Clade (1A.3.3.3)

Looking at the community graphs for Gamma Clade across all the layers shows that there definitely are different tightly knitted clusters developing within Gamma. Even though the
Figure 3.1: H1 HA Sb Epitope with user sequences
Figure 3.2: H1 HA Strain Ranking
dataset is small, gradual antigenic drift can be noticed. There were about two to four communities recorded by different layers (see Figures 3.3, 3.4, 3.5, 3.6, 3.7 and 3.8). With increase in dataset gradually to match the current amount of gamma sequences in the ISU VDL FLUture database we can notice either increase in the number of communities or actually combine to reduce the numbers. Also, it will be interesting to see change in community formation by building the dataset on a timeline. With 80 sequences and then further with 160 sequences it was noticed that all the layers were generating 4 communities which can be an indication of convergence of sub clades that needs to be studied from biological side to cross-confirm.

(a) partial graph with user epitope sequence.  
(b) Three communities detected.  

Figure 3.3: Sa Epitope Gamma

3.1.3 Analyzing user sequence against gamma strains based network

For our test case, when a couple of non-gamma sequences were passed to a gamma based network it was noticed that the ranking returned an empty set, see Figure 3.10 for sequences mapped against the partial graphs across layers. It implies that for this particular test scenario, the sequences were not part of any gamma strain communities across the network layers.
(a) partial graph with user epitope sequence.  
(b) Four communities detected.

Figure 3.4: Sb Epitope Gamma

(a) partial graph with user epitope sequence.  
(b) Two communities detected.

Figure 3.5: Ca1 Epitope Gamma
(a) partial graph with user epitope sequence.  

(b) Three communities detected.  

Figure 3.6: Ca2 Epitope Gamma

(a) partial graph with user epitope sequence.  

(b) Two communities detected.  

Figure 3.7: Cb Epitope Gamma
(a) partial graph with user epitope sequence.  

(b) Four communities detected.

Figure 3.8: Strain Epitope Gamma
Figure 3.9: Gamma Result
Figure 3.10: Gamma Network with Non-Gamma sequences
CHAPTER 4. SUMMARY AND DISCUSSION

Now, after gathering the results its time to get back at the original hypotheses and discuss them. It was also noted that the amount of computational time taken by the strain sequence layer to build distance matrix was more than 95% of the total time taken to generate all the matrices. Thus, its very important to figure out a plan for optimizing that step.

4.1 Introduction

Our hypotheses were whether the use of epitope sites and epitope in conjunction with the strain data can be used to find important strains and immune epitopes within the protein. Also, whether an antigenic drift can be inferred from the epitope network.

4.1.1 Epitope result different from sequence

It can be noticed by comparing epitope graphs and results vs sequence’s to see that epitope and sequences can end up giving same results, but also that if two strains are identified as very different by sequence analysis epitope analysis result can point out that they are in reality very similar instead. This has to be further build on quantitatively with different test cases.

4.1.1.1 Use strains as vaccine candidates

The different internal common nodes noticed in the egonet graphs are some of the important strains per layer that can be used as a candidate for IAV vaccine development.

4.1.2 Antigenic Drift through gamma clade study

We noticed different number of communities sprouting within each layer of the network. Though, it will need a much larger dataset to identify how the gamma clade has evolved with time.
4.1.2.1 Identifying new clades

This method if applied to other clades would definitely give us a deeper insight into how those clade are evolving with time.

4.2 Future Work

Compare the rank results against BLAST results made on the same mini dataset to cross-verify the results further and quantify them. Also, it needs to be evaluated how many communities would be optimal as discussed by Newman and Girvan (2004). Creation of dendrograms from the community structure results would give a better pictorial representation of the communities. Apart from scaling up the network and make code work with parallel processing, there is a need to apply distribution functions to find better \textit{cutVal’s}. It need not be mentioned but BLOSUM62 based sequence similarity to generate distance matrix and automated reports will be helpful to compare with levenshtein based distancing results. As we mentioned in section 3.1.1, the naive ranking isn’t very helpful, so instead use pairwise \textit{Edgebetweennessvalue} across different network layers to create a weighted ranking that will be much more representative of the overall similarity.
BIBLIOGRAPHY


