CORNELL TECH

COVID Variants Protein Clustering

Final Report

Ze Yuan Li, Youan Lu, Aman Prasad, Jiadong Xu

`zl344, yl889, ap798, jx248`

CS - 5112 Algorithms and Data Structures

January 6, 2022

Contents

1 Motivation **1**

2 Data **1**

2.1 Data collection 1

2.2 Pre-processing 2

3 Methods **2**

3.1 Distance Measurement Algorithms 2

3.1.1 Euclidean Distance 2

3.1.2 Ratcliff-Obershelp(RO) Algorithm 3

3.1.3 Hamming Distance 3

3.2 Clustering Protein Sequence based on the DBSCAN algorithm 3

3.3 Distance matrix with Hierarchial clustering 5

3.4 Distance matrix with TSNE (T-distributed Stochastic Neighbor Embedding) 5

4 Results **6**

4.1 K-Means 6

4.2 DBSCAN 6

4.3 Distance matrix with Hierarchical clustering 9

4.4 Distance matrix with TSNE 10

5 Conclusion and Future Work **11**

1 Motivation

In the past two years, an outbreak of Coronavirus from the Huanan Seafood market has turned the world upside down. Humanity has since been battling with this terrifying pandemic. While promising treatments and vaccinations have been made widely available, the war with COVID endures due to constant evolution of the virus. For example, while the original strain can be protected by the MRNA vaccines fairly well, later variants (Delta and Omicron) have begun to escape the wall of vaccine protection.

Thus, variants need to be identified and assessed periodically whether they pose new threats. In this paper, we aim to find a effective and efficient way of identifying which variant a spike protein belongs to using only amino acids sequence with no prior biological knowledge. We believe our work can contribute to identifying variants swiftly and drive public health response when potential new variants are discovered.

2 Data

2.1 Data collection

We collected the six COVID 19 variants' spike protein sequence: Alpha, Beta, Delta, AY42, Gamma and Omicron. The data was downloaded from NCBI SARS-CoV-2 Resources and for each variant we collected 200 protein sequences (Aside from Omicron, which we had 8 sequences) . Each protein sequence contained ∼1270 amino acids and thus we have strings of 1270 - 1275 length of characters for each data point.

2.2 Pre-processing

We used a python package called BioPython that was built to for .fasta format files. This package allowed us to neatly organize the Pangolin, and sequences. In addition, it could operate basic operation on sequences.

To perform clustering on protein sequence, we first tokenized the protein sequence by n-grams using CountVectorizer (a package built into scikit-learn). We experimented with 3, 6, and 9-grams to determine which is the most effective in generating distinct vectors for each variant. Through our experimentation, we were able to determine that 9-grams were the most expressive and led to the most distinct clusters. In the context of protein sequences, n-grams transforms the string into frequency vectors for text analysis. For a COVID-19 Alpha variant protein sequence ”MFVFLVLLPLVS...”, Count Vectorizer transforms to columns of ”MFVFLV” and ”LLPLVS” with row of whether the token exist in the sequence.

3 Methods

3.1 Distance Measurement Algorithms

3.1.1 Euclidean Distance

As a baseline measure of distance between two protein sequences, we used the Euclidean distance metric. We compute the Euclidean distance by simply taking two 9-gram vectors and computing the distance. This is a simple metric that is quick to calculate that we used in our initial experiments, namely with algorithms such as K-Means and DBSCAN.

3.1.2 Ratcliff-Obershelp(RO) Algorithm

The Ratcliff-Obershelp is also known as Gestalt Pattern matching is a string-matching algorithm that was developed in 1983 by John Ratcliff and John Obershelp. The score is calculated with the following equation:

$$D_{ro} = \frac{2K_m}{|S_1|+|S_2|}$$

where $|S_1|$ and $|S_2|$ are the length of the two strings being compared. K_m is the matching characters between the two strings determined by the longest common sub-string ran recursively on the non-matching half. As a result, the run time of this algorithm can be quite lengthy especially for longer inputs[3]. The final distance measure is 1 - D_{RO}.

3.1.3 Hamming Distance

Hamming distance was a similarity measure covered in class. The distance is determined by the number of positions that are different between the two strings. This algorithm is much faster to run than Ratcliff-Obershelp. The distance is simply the Hamming distance.

3.2 Clustering Protein Sequence based on the DBSCAN algorithm

The third algorithm we explored is DBSCAN. DBSCAN is one of the most commonly used density based clustering algorithms. The algorithm recognizes the clusters based on the points that are closely packed and have higher density than points out of the cluster. Unlike K-means algorithm where each partition is equivalent to a Voronoi diagram and the cluster shape is always convex, there is no restriction in shape of the DBSCAN clusters. We are interested in exploring how the difference will affect the clustering on protein sequence.

The DBSCAN algorithm determines the clusters in the following way: it first starts with a random points p and collect all the other points that are density-reachable. Point x is density-reachable from point y if the points distance is within and y is within a cluster(core point) ε or a chain of points $p_1 ... p_n$ and $p_1 = x$ and $p_n = y$ and thus p_{n+1} is density-reachable from p_n.[1]. If the number of points that are classified as density-reachable are more than minPts, a cluster will be formed. Otherwise it will be temporarily labeled as noise and may be revisited later. The algorithm continues to classify each non-core point to a nearby cluster within ε or to noise.

With the 9-grams data, we performed parameters tuning on the algorithm. There are three important parameters need to consider when building up clusters in DBSCAN algorithm:

1. the minimum number of points in a cluster, minPts
2. the choice of distance function
3. the maximum distance between two points to be considered as a cluster ε

In each cluster, within a given radius, the number of points, which is also called density, needs to exceed the given threshold of the minimum. The distance function determines the shape of the cluster and the ε determine the radius and size of the cluster. We estimate the parameters using the following steps.

1. minPts: We know that there should be three different clusters (Alpha, Delta, Gamma) and we go for the rule of $2 \times \textit{dimensions}$, which is 6 minPts.

2. distance function: We first measure the similarity of the matrix and discovered that the sequence matrix is binary list. Besides the most commonly used distance function, Euclidean distance, we also explore the Hamming distance in clustering the protein sequence.

3. ε: Choosing a suitable ε is very important in DBSCAN algorithm. If ε is too small, there would be too many clusters. But a large ε will merge more points than expected into a cluster. As the value of ε is closely related to the distance function, we adjust the value of ε based on each distance function.

3.3 Distance matrix with Hierarchial clustering

Two distance matrices were calculated using both Hamming Distance and the Ratcliff-Obershelp Algorithm. Distances were calculated between every i^{th} and j^{th} points, where $D(i, j) = D(j, i)$. Once the distance matrix was calculated, hierarchial/agglomerative clustering was used (with python library) to produce the clusters. To visualize, Dendrograms were used to show the clusters between the COVID samples.

3.4 Distance matrix with TSNE (T-distributed Stochastic Neighbor Embedding)

Similar with the above section, distance matrices were calculated with Hamming Distance and the Ratcliff-Obershlp Algorithm. Instead of using Hierarchial clustering, TSNE is used. TSNE first calculates use probabilistic models in higher dimensions to cluster similar points. In our case, it clusters points that are close to each other according to the distance. Next, TSNE brings the points to lower dimensions that can be helpful in visualization by comparing probability distribution using KL divergence.

4 Results

4.1 K-Means

Using the vectorized sequence data, we used clustering with K-Means as a baseline algorithm, as it is the most basic clustering algorithms out of the ones that we tried. In order to use K-Means, we tried using 3-grams, 6-grams, and 9-grams as the input to the Count Vectorizer.

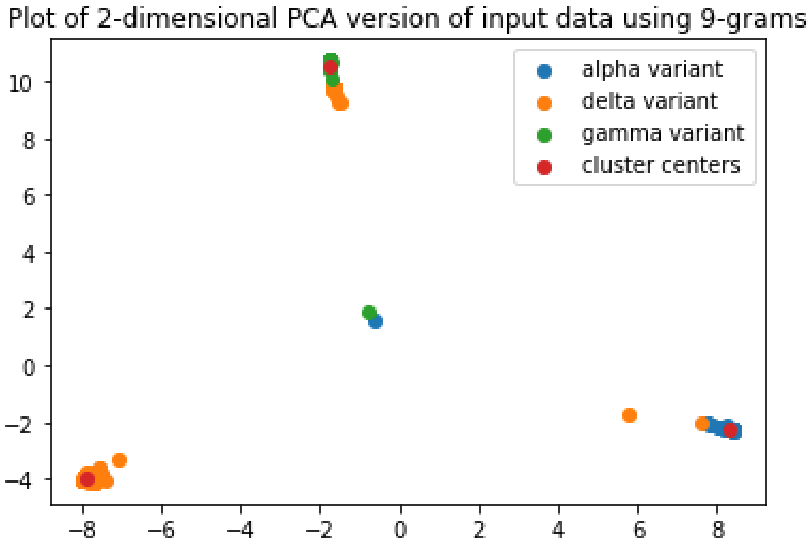


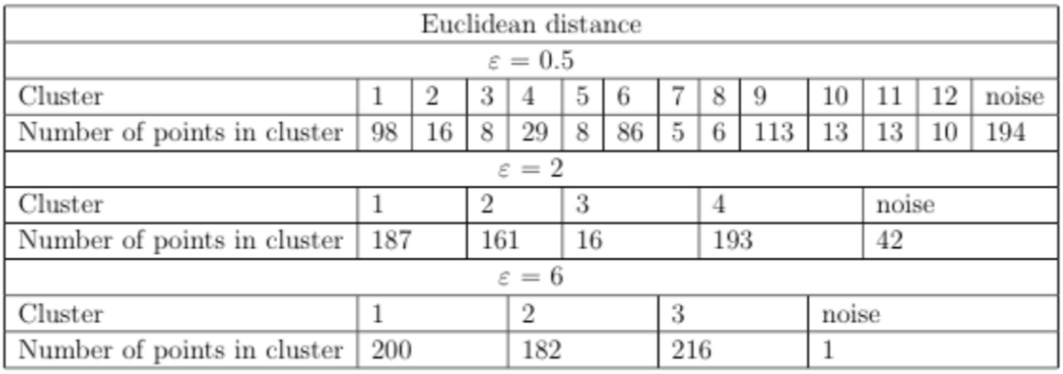
Figure 1: K-Means clustering result using 9-grams with three target clusters

This figure shows results that indicate that K-Means is able to effectively determine where to place cluster centers to separate the Alpha, Delta, and Gamma COVID variants. Because of its simplicity and effectiveness, we used this as a baseline algorithm to compare more complicated algorithms against.

4.2 DBSCAN

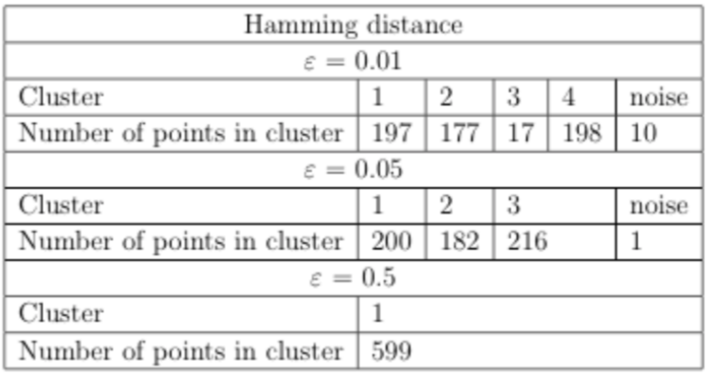
The data we used in DBSCAN is 200 Alpha, Delta, Gamma COVID variants. Referring to Figure 1 and Figure 2, we ran a different range of ε on each distance function. We discovered that both distance functions are able to converge the points to three clusters with one data point as a noise, and same number of points are allocated to a cluster when $\varepsilon = 2$ in Euclidean distance and $\varepsilon = 0.05$ in Hamming distance. We conducted further data exploring under this parameter setting. DBSCAN has a deterministic result, We discovered that the true value for the noise point is in Delta variant as it has slightly different sequence with the other sequence. It turned out that DBSCAN is able to target noise. Besides, there were 16 points of Delta variant are wrongly classified to Gamma variant. We realized that these sequences are the same and repetitive, which may signal a labelling mistake by NCBI. The misclassification can also be caused by Count Vectorizer we used for data pre-processing. Count Vectorizer only considered the occurrence of the terms and does not consider the permutation of each amino acids. A further improvement can be in finding a better way for pre-processing the protein sequence.

However, during the whole process, we realized that superior to K-means, DBSCAN did not require to know the number of clusters in the data. In more cases, there is no certain cluster value in a clustering problem. DBSCAN will be very helpful in these cases. It was not easy to determine proper ε and minPts in DBSCAN. Proper results can only be achieved with a valid understanding of the data and the scale. The ε and minPts value needed to be adjusted in every different distance function or size of dataset. For high dimensional data, it was harder to find a appropriate ε and further data exploring needed to be done to set up the model. Besides, if the data has a large difference in density, the choice of minPts will also be difficult as we want the small density data to be classified as one cluster but not merging into the other cluster.



Euclidean distance													
$\varepsilon = 0.5$													
Cluster	1	2	3	4	5	6	7	8	9	10	11	12	noise
Number of points in cluster	98	16	8	29	8	86	5	6	113	13	13	10	194
$\varepsilon = 2$													
Cluster	1	2	3	4	noise								
Number of points in cluster	187	161	16	193	42								
$\varepsilon = 6$													
Cluster	1	2	3	noise									
Number of points in cluster	200	182	216	1									

Figure 2: DBSCAN Clustering result with different ε in Euclidean distance



Hamming distance					
$\varepsilon = 0.01$					
Cluster	1	2	3	4	noise
Number of points in cluster	197	177	17	198	10
$\varepsilon = 0.05$					
Cluster	1	2	3		noise
Number of points in cluster	200	182	216		1
$\varepsilon = 0.5$					
Cluster	1				
Number of points in cluster	599				

Figure 3: DBSCAN Clustering result with different ε in Hamming distance

4.3 Distance matrix with Hierarchical clustering

We first began calculating the distance by using the entire viral RNA sequence (27k base pairs). However, applying RO turned out to be run-time nightmare; it took 5 hours one time to compute pairwise distance for 20 samples. RO run time is the slowest if two sequence are from different variants, faster if the two are from the same variant, and fastest when the two sequences are identical. Nonetheless as we tried to compare more base pairs, the distance matrix's size increased in quadratic order and the run-time became more problematic.

We tried several different approaches guided by scientific research on mutations of the spike protein[4]. This paper identified several regions on the spike protein where mutations are common. Specifically, majority of the mutations occurred along the $S1^D$ domain between amino acids 594-674. Utilizing a shorter sequence, we were able to achieve accurate classification of the Alpha and Delta strains. We were also able to drastically improve the run time with the a shorter string to process; reducing the string length by a factor of 100. However, we did not prefer this approach as it only captures mutation from certain region. Thus, other crucial changes can be missed and any inference from our method may be hard to be accepted by the scientific community.

Ultimately to address the run time, we decided to use Hamming distance. The results are very promising. In this following diagram, we can see that the variants are appropriately labelled according to the Dendrogram; Each color is its own variant. However, it can be very difficult to visualize. Hence we decided to use something that can better visualize the clusters.

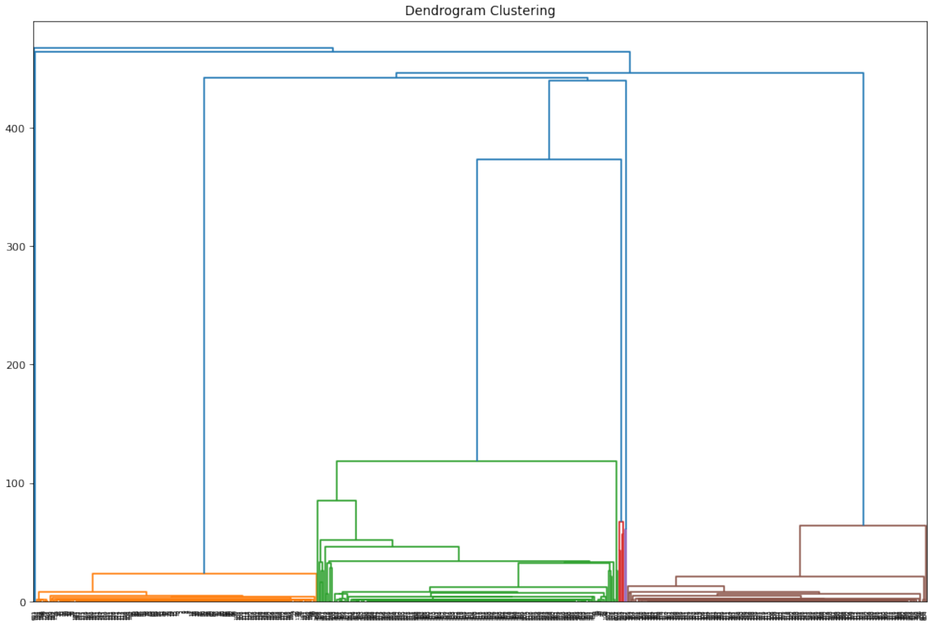


Figure 4: Dendrogram using Hierarchial Clustering

4.4 Distance matrix with TSNE

We wrote code in python to implement TSNE. To visualize, we set n_components to 2 which gave a two dimensional representation. The perplexity parameter (similar to number of neighbors) was set to the default 30. This gave clustering with clear boundaries.

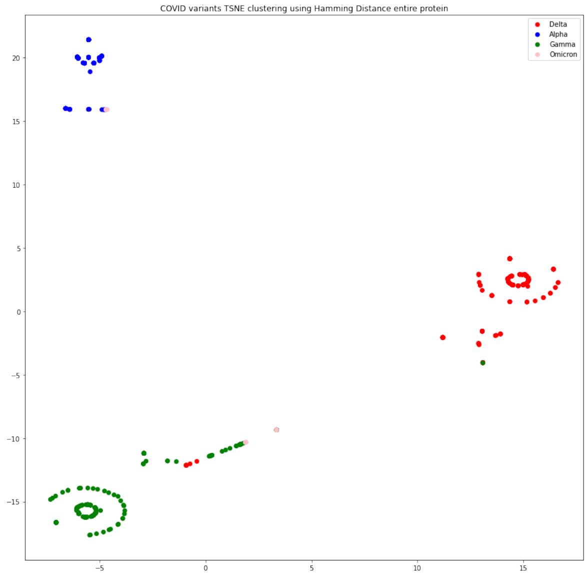


Figure 5: TSNE clustering with Delta, Alpha. Gamma and Omicron sequences

We also confirmed the **Omicron variant** was not present in previous samples:

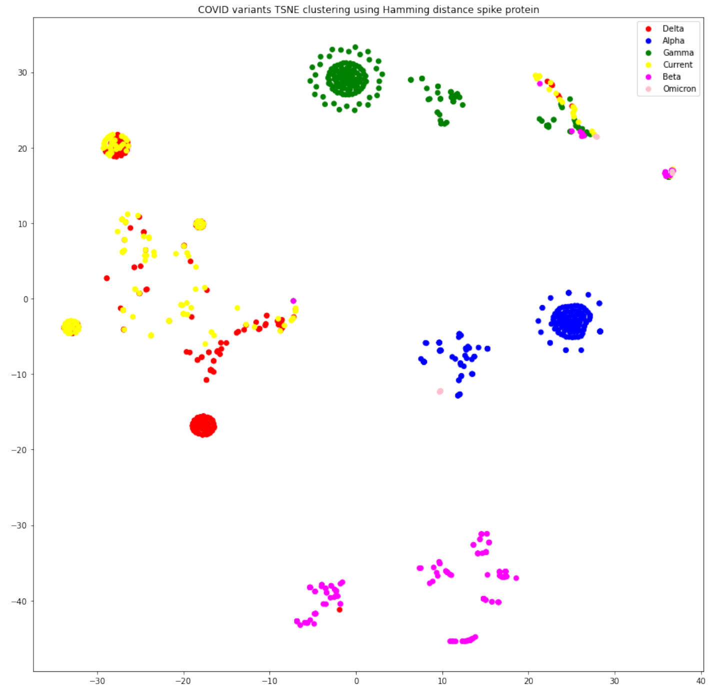


Figure 6: TSNE clustering with unlabelled sample (mid Nov 2021)

Thus, we have shown TSNE with Distance Matrix as a suitable way to cluster and visualize COVID variants.

5 Conclusion and Future Work

With the three methods (K-Mean, DBScan, and Distance Matrices), we were able to achieve high clustering accuracy with respect to the golden label in the NCBI database. For example, the DBSCAN algorithm returned the new clustering label for the COVID variants, we are able to confirm that the classification achieved 92% accuracy. We were also able to achieve a 92.5% similarity score between the labels generated by K-Means and the true variant labels. Finally, the distance matrix and TSNE approach involving

Alpha, Delta, and Gamma variants also achieved over 96% accuracy comparing to golden labels. Thus, we have demonstrated multiple efficient ways to cluster COVID variants.

In the future, we can incorporate continuous emergence of COVID variants to determine how they fit in within previous groupings. We would also try to use some more complex quantitative metrics to determine the effectiveness of each of the clustering algorithms. Finally, we hope to show case our approach to public health agencies and look for ways to end this pandemic ASAP.

References

[1] Shah, Glory H. “[PDF] an Improved DBSCAN, a Density Based Clustering Algorithm with Parameter Selection for High Dimensional Data Sets: Semantic Scholar.” Undefined, 1 Jan. 1970, https://www.semanticscholar.org/paper/An-improved-DBSCAN %2C-a-density-based-clustering-with-Shah/ e413ec9cb98febf87 0ac16d227e41c1a9485a5f6.

[2] A Density-Based Algorithm for Discovering Clusters in ... https://www.aaai.org/Papers/KDD/ 1996/KDD96-037.pdf

[3] Ratcliff/Obershelp Pattern Recognition, https://xlinux.nist.gov/dads/HTML/ ratcliffObershelp.html.

[4] L;, Guruprasad. “Human Sars Cov-2 Spike Protein Mutations.” Proteins, U.S. National Library of Medicine, https://pubmed.ncbi.nlm.nih.gov/33423311/.