GENUS LEVEL CLASSIFICATION OF SARS-COV-2 USING INTRINSIC GENOMIC SIGNATURES

A PREPRINT

Aadharsh Aadhithya A Anirudh Edpuganti Onteddu Chaitanya Reddy Pillalamarri Akshaya

Dr.Sachin Kumar S Dr.Soman K P

ABSTRACT

Disruptions due to novel pathogens are inevitable. The recent pandemic due to SARS-CoV-2 is an indispensable example of this very fact. This paper aims in providing a genus-level idea of a novel pathogen of its plausible genetic origins. In this paper, several Machine Learning techniques are used for the classification of SARS-CoV-2 at the genus level. This shall also give ideas on the origins of SARS-CoV-2.

1 Introduction

The causal pathogen for the recent Covid-19 pandemic is found to be a member of *Coronaviridae*. The *Coronavirdae* family consists of 4 genera, namely *Alphacoronavirus* , *Betacoronavirus* , *Gammacoronavirus* , *Deltacoronavirus.* From phylogenetic studies, it has been found COVID-19 virus belongs to the sub-genus *Sarbecoronavirus* of the *Betacoronavirus* genus.

Typical phylogenetic analyses are done using alignment and annotation-based methods. Analyzing thousands of genomes using computational methods could be computationally taxing. Recent democratization of computing power means increased access to alignment-free methods for phylogenetic analysis has been proposed.

This paper tries to build machine learning models in order to classify a novel pathogen at a genus level. The same techniques can be extended for higher-level classifications as well. Achieving high accuracies at a genus level means the machine learning models are learning to discriminate granular details well.It can be said with conviction that higher-level classifications would involve many distinct discriminations, hence our models can be extended to higher-level classifications as well.

The rest of the paper is structured as follows. Section 2 Describes the methods and techniques used, Results and Discussion are in Section 3.

2 Methodology

Genomes of Embecovirus, Sarbecovirus, Nobecovirus, Merbecovirus are collected from NCBI and Virus-Host-DB. The Dataset consists of 48 Embecovirus,18 Merbecovirus, 10 Nobecovirus, 46 Sarbecovirus Genomes. Accession numbers for these genomes are available in the Appendix.

Genomes are preprocessed before training machine learning models. The genomes must be converted to a numerical representation. We chose to use Chaos Game Representation(CGR).CGR is said to be incredibly useful as a genomic signature [cite]. It uncovers kmer patterns in a genome in a visual manner. The resulting CGR is an image. The image is then flattened. We take Fast Fourier transform of the flattened image and the magnitude spectrum of the flattened image can be considered as one feature vector.

Pearson Correlation Coefficient(PCC) between all the magnitude spectra of the numerical representations of the genomes are calculated and a distance matrix is formed. Dataset is visualized using Multidimensional Scaling(MDS)

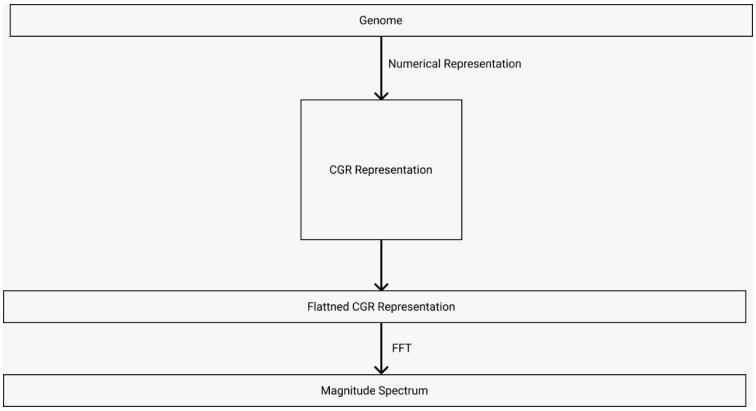

Figure 1: steps involved in preprocessing

and t-SNE. The columns of the distance matrix can be viewed as the distance of one genome from all other genomes. Columns of the distance matrix are also used as another feature vector.

Machine Learning models, Linear SVM, Quadratic SVM, Fine KNN, Linear Discriminant, and ensemble methods such as Subspace Discriminant, subspace KNN are trained first using magnitude spectra as a feature vector and then using the columns of distance matrix as a feature vector. Accuracy is obtained through 10-fold cross-validation and the results are then compared.

2.1 Chaos Game Representation

We have seen a lot of patterns that are complex structures and seem similar across different scales. These structures are termed Fractals. Chaos game is an algorithm by which, one could create fractals with simple governing rules. The famous Sierpinski triangle is an example of a Chaos game. Although it was randomly generated, we have seen evident patterns forming out in the case of the tri-nodal chaos game. Quadri-nodal chaos game yields a uniformly filled square. If the quadri-nodal chaos game isn't truly random, we can expect to see some patterns in quadri-nodal chaos games. Chaos game representation is a graphical representation of a genomic sequence. This is one of the unique techniques to convert the one-dimensional genomic sequence into a graphical image. Visual patterns in quadri-nodal chaos games with each node corresponding to one of the nucleotides - A, C, G, or T reveals interesting intrinsic signatures of the genome such as k-mer frequencies. Binning the Chaos Game Representation into $2^k \times 2^k$ bins and counting the number of points in each bin, gives us counts of all the unique k-mers present in the genome. This is the Frequency Chaos Game representation(FCGR).

2.2. Pearson Correlation Coefficient (PCC)

Pearson correlation coefficient is a measure of cosine similarity between two mean adjusted vectors. The positive values of the Pearson correlation coefficient denote positive linear correlation and similarly, negative values of the Pearson correlation coefficient denote the negative linear correlation. If the Pearson correlation coefficient is zero it indicates there is no linear correlation between two vectors. Strong linear correlation occurs when the value of the Pearson correlation coefficient is closer -1 or 1. Pearson Correlation Coefficient between two vectors $\vec{X}$ and $\vec{Y}$ can be given as

$$r_{XY} = \frac{\sum_{i=0}^{p-1} \left(X_i - \bar{X}\right)\left(Y_i - \bar{Y}\right)}{\sqrt{\sum_{i=0}^{p-1} \left(X_i - \bar{X}\right)^2} \times \sqrt{\sum_{i=0}^{p-1} \left(Y_i - \bar{Y}\right)^2}} \tag{1}$$

2.3. Fourier Transform

Discrete Fourier transform is applied to the numerical representation of the genome to get the magnitude spectra. The Frequency Chaos game representation(FCGR) with k=7 is flattened and passed through Fourier transform.

$$F(j\omega) = \sum_{k=0}^{N-1} f[k]e^{-j\omega kT} \tag{2}$$

Magnitude spectrum is obtained from the resulting sequence.

2.4. Multidimensional Scaling

Multidimensional scaling is a dimensionality reduction method, primarily used to visualize large datasets. Classical Multidimensional Scaling tries to preserve distances of vectors in higher dimensional space while projecting into typically a 2 or 3-dimensional space. This requires a pairwise distance matrix ($N \times N$) and produces $N \times Q$ coordinate matrix, where Q is typically 2 or 3. However, preserving distances and projecting might not be sufficient because of the curse of dimensionality. For better visualization, we shall also use t-distributed Stochastic Neighbour Embeddings.

2.5. t-Stochastic Neighbor Embeddings

t-Stochastic Neighbor Embeddings is a nonlinear dimensionality reduction technique, developed as an improvement to methods of stochastic neighbor embeddings. A similarity measure between instances in high and low dimensional space is optimized using a cost function. Similarity measures are obtained both in high dimensional space and its map in low dimensional space from probability distributions($p_{ij} \rightarrow$ similarity between point x_i and x_j in high dimensional space and $q_{ij} \rightarrow$ similarity between point y_i and y_j in lower dimensional space). t-SNE learns a map from high dimensional data to low dimensional data such that p_{ij} and q_{ij} is as near as possible. This is accomplished using optimization . KL divergence in Equation 5 can be used as cost function.

$$p_{ij} = \frac{\frac{exp(-|x_i - x_j|^2)}{2\sigma_i^2}}{\sum_{k \neq i} \frac{exp(-|x_i - x_k|^2)}{2\sigma_i^2}} \tag{3}$$

$$q_{ij} = \frac{(1 + |y_i - y_j|^2)^{-1}}{\sum_k \sum_{l \neq k} (1 + |y_k - y_l|^2)^{-1}} \tag{4}$$

$$KL(P||Q) = \sum_{i \neq j} p_{ij} log(\frac{p_{ij}}{q_{ij}}) \tag{5}$$

2.6. SVM

SVM(Support Vector Machine) is one of the supervised machine learning models which is used to solve both classification and regression problems. The major advantages of SVM are more speed and better performance. Suppose we have some data points and each data point belongs to one of the two classes, our goal is to classify which class does the data point belongs to. Support vectors are the data points that lie closest to the decision surface (or hyperplane). This Algorithm chooses the optimal hyperplane in which the margin around the separating hyperplane is maximum. If two planes can be separated by a straight line it can be known as Linear SVM. If the planes that separate the classes cannot be separated by a straight line and takes the form of a curve can be said as Quadratic SVM.

2.7. Linear Discriminant Analysis (LDA)

Linear Discriminant Analysis is one of the techniques used to classify the data sets specifically in supervised learning. This method classifies the data based on different characteristics noted on different axes. If the number of characteristics or independent variables increases, classification based on those independent variables becomes tough. So, LDA approaches this more optimally by considering the importance of all the characteristics and reducing it to a single dimension. This is known as Dimension Reduction. So, this creates a new axis which is a mixture of all those characteristics specified.

LDA bases on two main criteria while creating the new axis
1. Maximize the difference between means of clusters
2. Minimize the distance between points of a cluster to its mean.

In other terms, it maximizes the separation between the data points of different clusters.

2.8. Fine KNN

The k- nearest neighbors (KNN) is a simple supervised learning technique that can be used to solve both classification and regression. This method is greatly used in Data Mining and Machine Learning algorithms due to its simple implementation. In the KNN classification, the decision is made using the information about the euclidean distances of 'K' most nearest neighbors which are lying closer to a data point.

$$\text{Euclidean Distance} = \sqrt{\sum_{i=1}^{k} (x_i - y_i)^2} \tag{6}$$

$$A = \{p_i : supremum(Neighbours(p_i)) \, \epsilon \, A\} \tag{7}$$

2.9. 10-Fold Cross-Validation

The problem in many cases is that we don't have enough data to train and test them with a different set of data. Hence, to train and test the data with the same dataset, we use the concept of cross-validation. In 10-fold cross-validation, the original sample is divided into 10 subsamples. One subsample among the 10 will be randomly chosen as the validation data for testing the model and the remaining subsamples will be considered as the training data. This continues 10 times and each of the subsamples is considered as the validation data for training the model at least once. Hence, 10 outcomes will be produced in this process and all the 10 are averaged in order to produce a single accuracy estimate. This method is advantageous since each subsample is used as a testing set exactly once and all the samples are used as the training set, the same number of times i.e nine times.

3 Results and Discussion

The dataset had been visualized by dimensionality reduction techniques such as Multidimensional Scaling and t-Stochastic Neighbour Embeddings. These techniques are used to project high dimension data to 2 or 3 components. This paper had projected the data from $R^{128\times128}$ to R^3.

Multidimensional scaling does a good job of projecting the data to 3 dimensions as several clusters are visible. Accuracy for 6 machine learning models was noted after 10 fold cross-validation and are as follows.

	ML Models	Accuracy(%)(Magnitude Spectrum)
1	Linear Discriminant	95.57
2	Linear SVM	98.80
3	Fine KNN	94.08
4	Quadratic SVM	99.18
5	Subspace KNN	98.67
6	Subspace Discriminant	99.10

	ML Models	Accuracy(%)(Distance Matrix)
1	Linear Discriminant	95.14
2	Linear SVM	98.43
3	Fine KNN	94.08
4	Quadratic SVM	99.67
5	Subspace KNN	98.67
6	Subspace Discriminant	99.10

It can be observed that both, Magnitude Spectrum as a feature vector and the columns of the distance matrix as a feature vector produces similar accuracies. Since we are required to calculate the distance matrix for each new genome, we picked the models with magnitude spectrum as a feature vector and deployed it **'dynamic' not supported in LaTeX**.

Out of the 6 Machine Learning models, Highes accuracy is observed in Quadratic SVM with 99.67%. It also correctly classifies SARS-CoV-2 genome to the Sarbecovirus sub-genome.

4 Conclusion

This paper analyzed the Covid-19 data using 6 Machine Learning models and then cross-validated them using the 10-fold cross-validation technique and obtained accuracies for each of the six Machine Learning models.

From the results obtained, Fine KNN is shown to be the least accurate Machine Learning model among the six models used with an accuracy of 94.08% in both the cases of Magnitude Spectrum and Distance Matrix as the Feature Vectors.

Also, Quadratic SVM is considered to be the best Machine Learning model from the 6 models considered with an accuracy rate of 99.18% in the case of Magnitude Spectrum as a feature vector and with an accuracy rate of 99.67% when Distance Matrix is used as a Feature Vector.