Hierarchical Clustering on RNA Dependent RNA Polymerase using Machine Learning.

GUDIPATI.PAVAN KUMAR*

* Independent Researcher, Hyderabad, India.

Abstract: RNA Dependent RNA Polymerase (RdRP) catalyzes the replication of RNA from an RNA template and is mostly found in Viruses. We have collected over 161 viral RdRP FASTA Sequences from the NCBI protein database using python script. Each of these sequences was transformed with TfidfVectorizer using sklearn module, with the "one Letter" word, because each Letter belongs to one Amino acid. These transformed data were sent to Hierarchical clustering using scipy library and visualized data using Dendrogram. These Machine Learning technique is able to classify or segment similar RdRp into one cluster. Each of these clusters was tested for their multiple sequence alignment with COBALT of NCBI. We observed that these clusters predicted similar RdRP among various viruses. These techniques can be further improved to segment or classify various proteins. These Machine Learning or Artificial Intelligence techniques need more improvement in their algorithms to solve genomics and proteomics.

Keywords: RNA Dependent RNA Polymerase, RdRP, Machine Learning, *Hierarchical* Clustering, sklearn, scipy, TfidfVectorizer, Dendrogram, Biopython

Repository: Programs and Datasets : https://github.com/DSPavan/RdRP

I. INTRODUCTION

RNA-dependent RNA polymerase (**RdRp**) is an enzyme that catalyzes the replication of RNA from an RNA template. RNA dependent RNA polymerase (RdRp) of SARS COV-2 is known to mediate replication of the viral genome and its propagation inside host cells [1]. Several RNA containing Viruses have this enzyme. ***Hierarchical clustering***, also known as *hierarchical cluster analysis*, is an algorithm that groups similar objects into groups called *clusters*. The endpoint is a set of clusters, where each cluster is distinct from each other cluster, and the objects within each cluster are broadly similar to each other. [2,3]. NCBI Protein database is a collection of sequences from several sources, including translations from annotated coding regions in GenBank, RefSeq and TPA, as well as records from SwissProt, PIR, PRF, and PDB. [4]. Python is a programing language widely used in Machine Learning, and they have ***sklearn*** [5] and ***scipy*** [6] modules. Main aim of this experiment is to test, can we cluster similar RdRP with help of Machine Learning / Artificial Intelligence techniques?

II. MATERIAL & METHODS

A) **Collection of RdRP**: Primarily this data collected from NCBI website, under Protein Database. These data can be collected manually from this database, using search option. We can also Biopython modules to get all these data from programmatically. I have extracted RdRP data using Biopython [7] with Entrez [8] module, specifically Amino acid sequences length between 600 to 3500 and excluded partial, probable, putative types from the list, so that we will get clean records. I have used kaggle notebook [9] with Python, which have preinstalled biopython libraries. I have collected 490+ RdRP from various organisms, mostly Viruses. After extraction of data, manually curation is done, to remove duplicates or identical proteins. After final curation we have dataset of 161 RdRP proteins.

Biopython CODE:

```
from Bio import Entrez
Entrez.email = "example@example.org"
```

```
searchResultHandle = Entrez.esearch(db="protein", term= "RNA-dependent RNA polymerase[Title] AND RNA-dependent RNA polymerase[Protein Name] AND 600:3500[Sequence Length]  NOT  ( partial OR probable OR TPA_asm OR pdb OR like OR eukaryotic-type OR putative OR TPA OR mitoviral OR like)", retmax=500)

searchResult = Entrez.read(searchResultHandle)
ids = searchResult["IdList"]
#print(ids)
handle = Entrez.efetch(db="protein", id=ids,  retmax=500, rettype="fasta", retmode="text")
record = handle.read()
# The output file will be saved under Dataset/output section
out_handle = open('RdRP3.fasta', 'w')
out_handle.write(record.rstrip('\n'))
#print(record)
```

B) **TfidfVectorizer:** Each Fasta sequences of RdRP was transformed with TfidfVectorizer method with in sklearn module. TfidfVectorizer, Convert a collection of raw documents to a matrix of TF-IDF features. In this experiment, we specifically tried with single letter word, so that, each amino acid in Fasta sequence can be transformed properly with above matrix.

#code

```
from sklearn.feature_extraction.text import TfidfVectorizer
vectorizer = TfidfVectorizer(stop_words={'english'}, analyzer='char', ngram_range=(1,1), min_df=1,max_df=1.0)
```

C) **Using Kmeans:** Number of optimum clusters were identified with Elbow technique and silhouette coefficients. Optimum Cluster to get all 161 Fasta sequences was 10 to 20 clusters.

D) **Hierarchical Cluster: After** TfidfVectorizer, this data clustered with Linkage type "ward". Then it is converted into Dendogram tree. After verifying dendrogram results, we can observe 8 major clusters, with color_threshold=0.14, each of these can be given unqiue color, so that, we can analyze each cluster for further analysis. You can vary color threshold, so that, you can verify different cluster sizes. I have kept this value (0.14), so that we can optimum size of clusters for analysis. With Cluster size-15, all 161 sequences clearly assigned to specific cluster.

#code

```
Zx = linkage(AAVector.toarray(), 'ward')
from scipy.cluster.hierarchy import dendrogram, fcluster, leaves_list, set_link_color_palette
set_link_color_palette([ "green","red","blue","orange","black","brown","Teal","Magenta"])

dend = dendrogram(Zx, color_threshold=0.14,leaf_font_size=12,labels=fheaderList, orientation='right', distance_sort='descending',truncate_mode='level',no_plot=True)
```

E) Extraction of Each Cluster for Further Analysis: We will get complete Denogram from above code, We need to extract, each color Cluster, using fcluster module, and we can extract list of fasta sequences of RdRP , which are grouped in same cluster.

#code

```
assignments = fcluster(linkage(AAVector.toarray(), method='ward'),0.14,'distance')
cluster_output = pd.DataFrame({'RdRP organism':fheaderList , 'cluster':assignments})
# you can modify  1 to 8 below, to get different clusters
newDF =  cluster_output['cluster']==1
```

In Results, you can get like this below

Eg: QXU64010.1 [Tomato spotted wilt orthotospovirus]

F) **Constraint-based Multiple Alignment Tool of NCBI:** Each of these clusters, which we extracted from above, was given as input (Accession Number is sufficient) into COBALT (Constraint-based Multiple Alignment Tool of NCBI) [10, 11]. After alignment, set Conservation Setting as "Identity". We collected, near to cluster branch as in dendrogram tree results, and send again for alignment, and we got below results.

III. RESULTS AND DISCUSSION

Each of these clusters sequences are aligned with COBALT- (Constraint-based Multiple Alignment Tool) with Conservation Setting as Identity.

A. Green Cluster

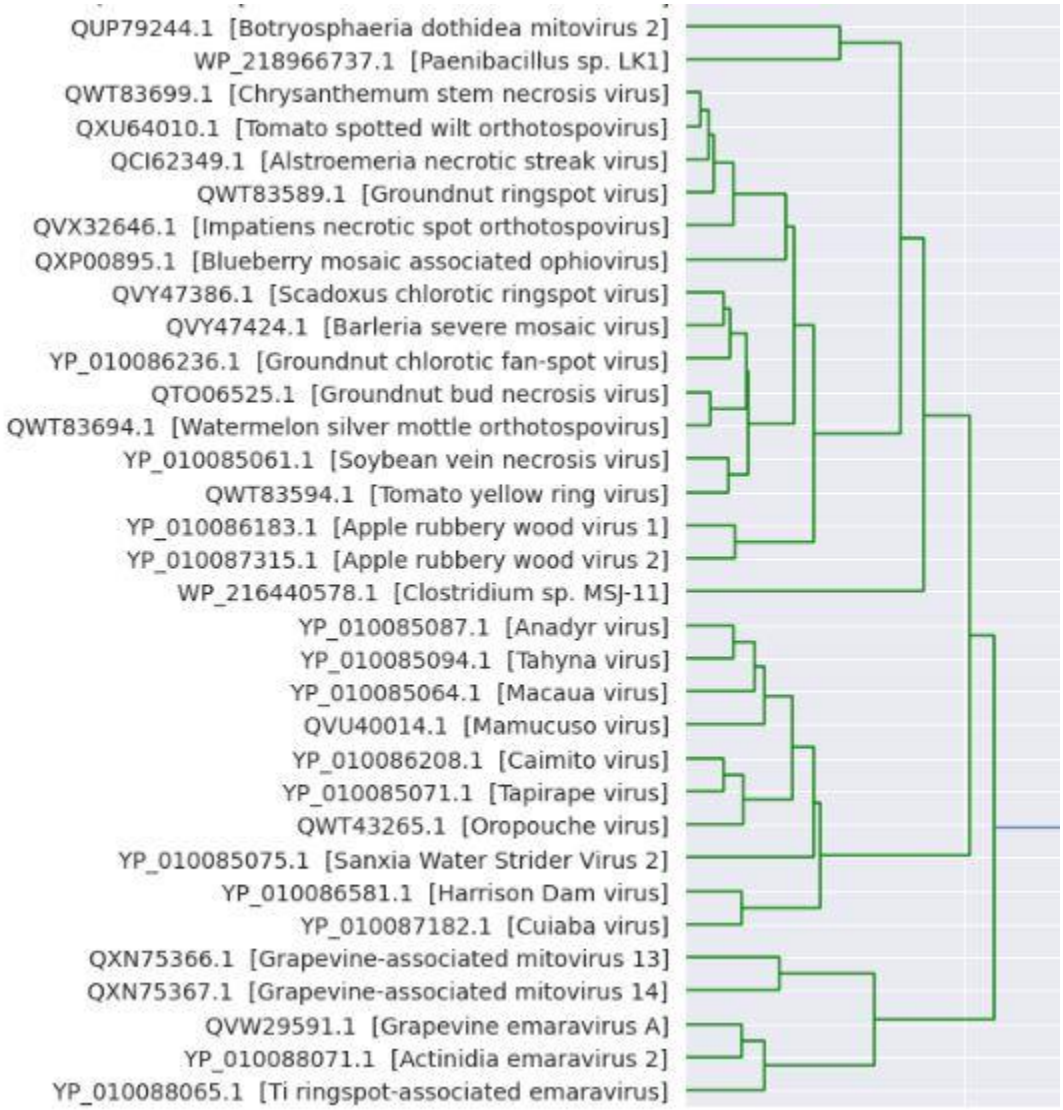

Fig. 1: Green Cluster.

Table 1: Green Cluster (Cluster No: 1) : Highly matching cluster

Set	Accession No.	Virus
A	QXU64010.1	Tomato Spotted Wilt Orthotospovirus
A	QWT83699.1	Chrysanthemum Stem Necrosis Virus
A	QWT83589.1	Groundnut Ringspot Virus
A	QCI62349.1	Alstroemeria Necrotic Streak Virus
B	QWT83694.1	Watermelon Silver Mottle Orthotospovirus
B	QTO06525.1	Groundnut Bud Necrosis Virus
C	QVW29591.1	Grapevine Emaravirus A
C	YP_010088071.1	Actinidia Emaravirus 2

B. Red Cluster

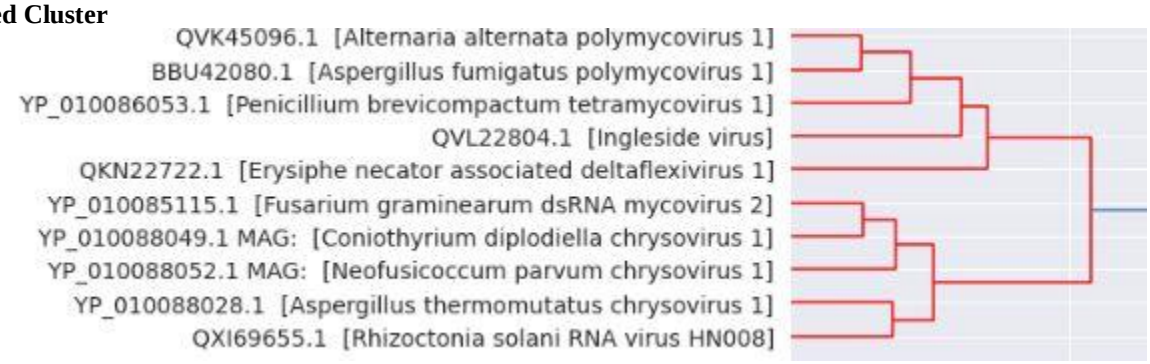

Fig. 2: Red Cluster.

Table 2: Red Cluster (Cluster No: 2): Highly matching cluster

Set	Red	Virus
A	BBU42080.1	Aspergillus fumigatus polymycovirus 1
A	QVK45096.1	Alternaria alternata polymycovirus 1
B	YP_010085115.1	Fusarium graminearum dsRNA mycovirus 2
B	YP_010088049.1	Coniothyrium diplodiella chrysovirus 1

C. Blue Cluster:

Table 3: Blue Cluster (Cluster No: 3) Highly matching cluster

Set	Accession No.	Virus
A	QWB49515.1	Carnation Italian Ringspot Virus
A	QVX32680.1	Cymbidium Ringspot Virus
A	QVX32675.1	Tomato Bushy Stunt Virus
B	QWC36209.1	Pelargonium Necrotic Spot Virus
B	QWC36204.1	Pelargonium Leaf Curl Virus

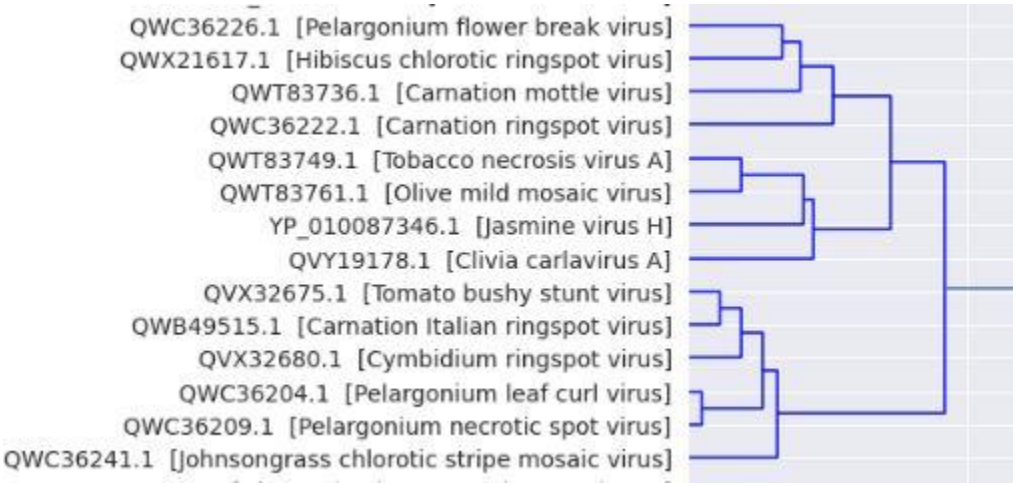

Fig. 3: Blue Cluster.

D. Orange Cluster:

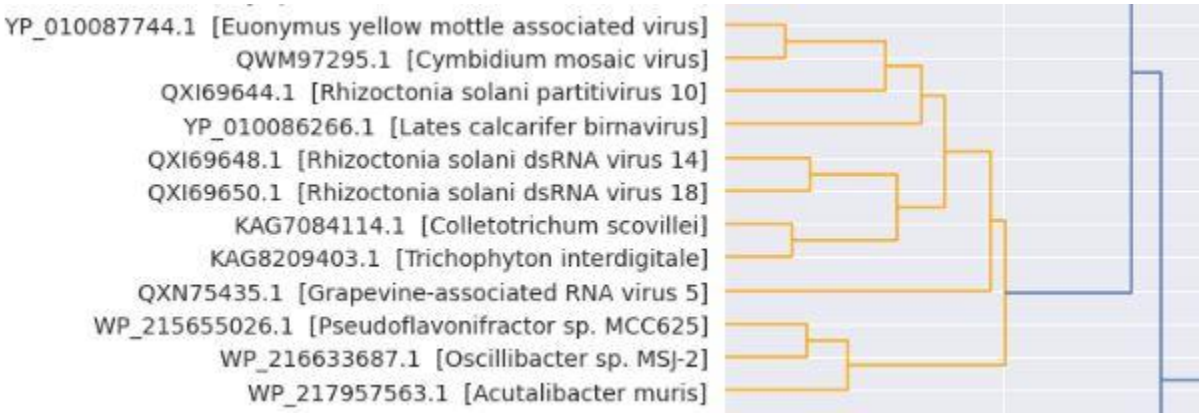

Fig. 4: Orange Cluster.

Table 4: Orange Cluster (Cluster No: 4): Highly matching cluster

Set	Accession No.	Virus
A	QWM97295.1	Cymbidium mosaic virus
A	YP_010087744.1	Euonymus yellow mottle associated virus
B	QXI69650.1	Rhizoctonia solani dsRNA virus 18
B	QXI69648.1	Rhizoctonia solani dsRNA virus 14
C	WP_216633687.1	Oscillibacter sp. MSJ-2
C	WP_215655026.1	Pseudoflavonifractor sp. MCC625
D	KAG8209403.1	Trichophyton interdigitale
D	KAG7084114.1	Colletotrichum scovillei

E. Black Cluster:

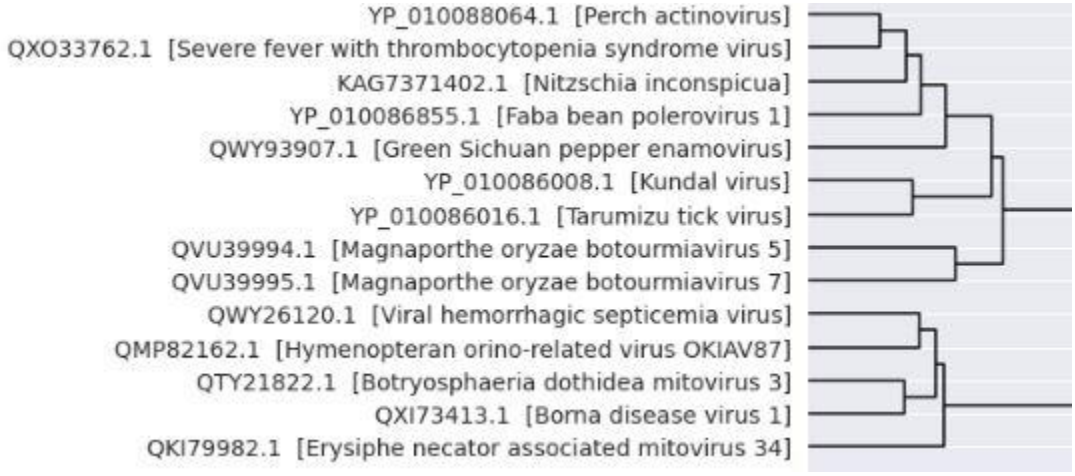

Fig. 5: Black Cluster.

Table 5: Black Cluster (Cluster No: 5) : Highly matching cluster

Set	Accession No.	Virus
A	YP_010086016.1	Tarumizu Tick Virus
A	YP_010086008.1	Kundal Virus

Note: There is no good match with other sequences, and huge difference is observed, probably because of long distance between them.

F. Brown Cluster:

Table 6: Brown Cluster (Cluster No: 6) : Highly matching cluster

Set	Accession No.	Virus
A	QXJ13539.1	Mokola Lyssavirus
A	QIK01206.1	Rabies Lyssavirus
A	YP_010086774.1	Taiwan Bat Lyssavirus

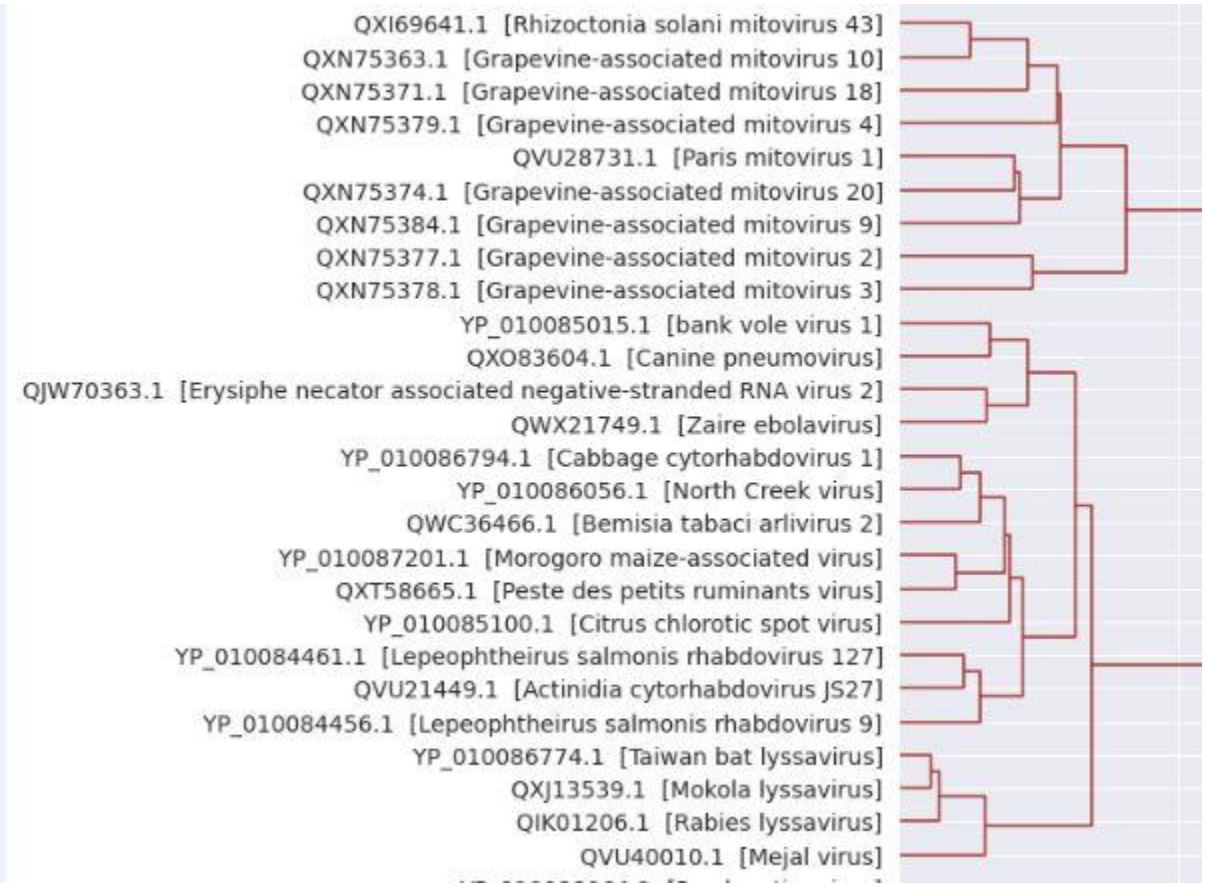

Fig. 6: Brown Cluster.

G. Teal Cluster:

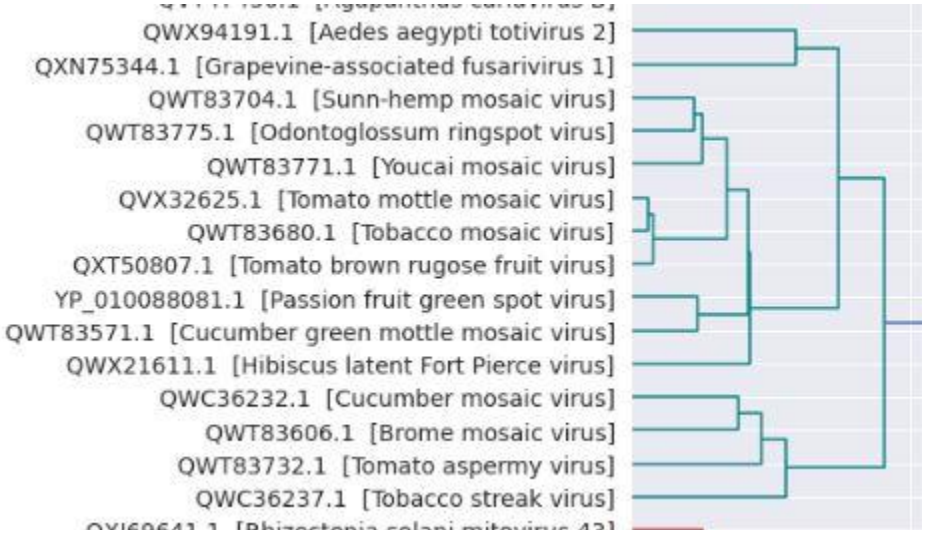

Fig. 7: Teal Cluster.

Table 7: Teal Cluster (Cluster No: 7) : Highly matching cluster

Set	Accession No.	Virus
Set A	QWT83775.1	Odontoglossum Ringspot Virus
Set A	QWT83771.1	Youcai Mosaic Virus
Set A	QWT83704.1	Sunn-Hemp Mosaic Virus
Set B	QXT50807.1	Tomato Brown Rugose Fruit Virus
Set B	QWT83680.1	Tobacco Mosaic Virus
Set B	QVX32625.1	Tomato Mottle Mosaic Virus

H. Magneta Cluster:

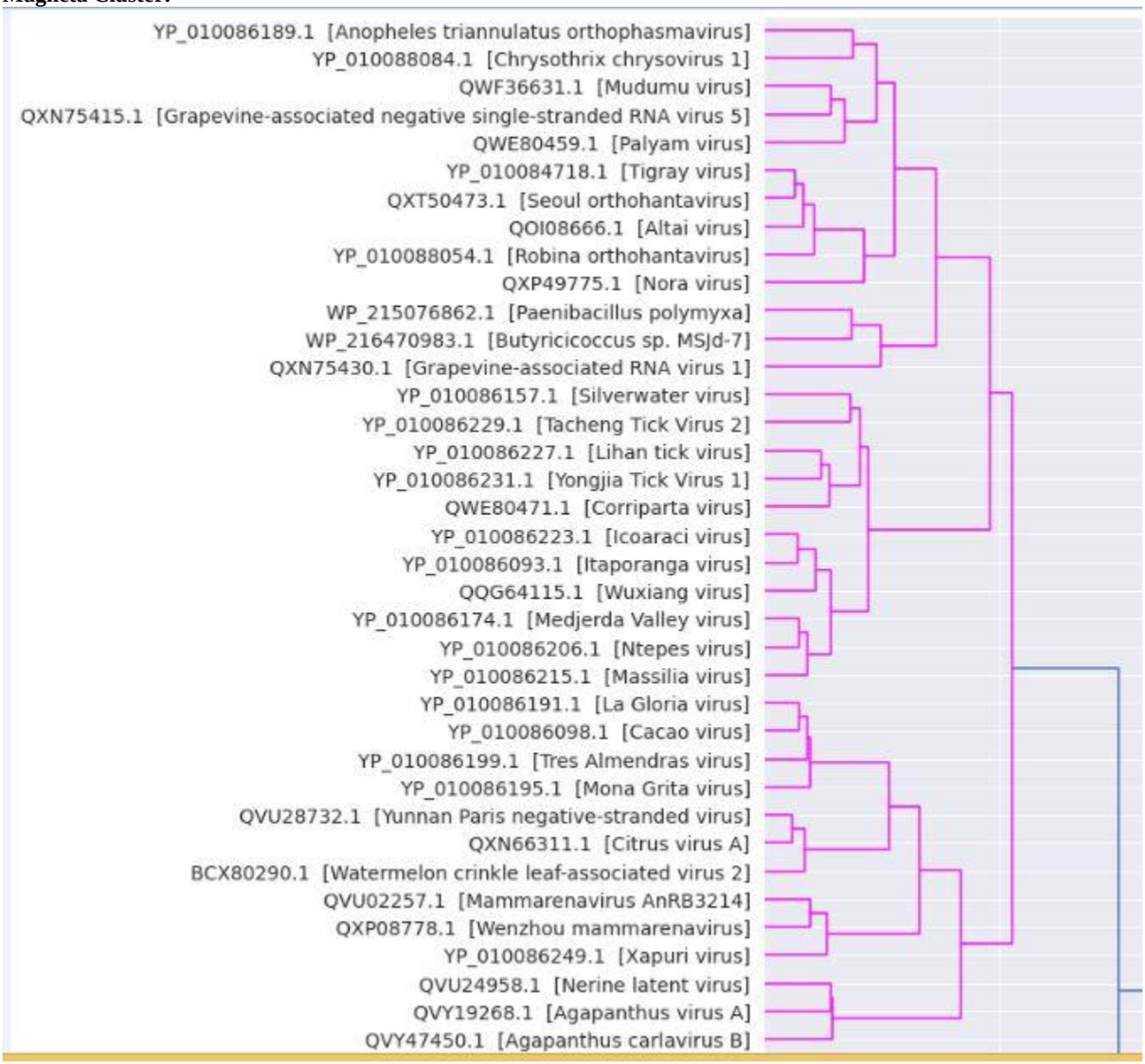

Fig. 8: Magneta Cluster.

Table 8: Magneta Cluster (Cluster No: 8): Highly matching cluster

Set	Accession No.	Virus
Set A	YP_010086098.1	Cacao Virus
Set A	YP_010086191.1	La Gloria Virus
Set B	YP_010086093.1	Itaporanga Virus
Set B	YP_010086223.1	Icoaraci Virus
set C	QXP08778.1	Wenzhou Mammarenavirus
Set C	QVU02257.1	Mammarenavirus Anrb3214

Constraint-based Multiple Alignment Tool output with Teal Cluster Set B as follows:

Fig. 9: COBALT output for Teal Cluster Set B

IV. CONCLUSIONS
V.

Hierachical clustering technique is able to identify similar groups based on Protein sequence. We able to identify similar groups eg.1: Tomato spotted wilt orthotospovirus, Chrysanthemum stem necrosis virus, Groundnut ringspot virus, Alstroemeria necrotic streak virus another example Rhizoctonia solani RNA virus HN008, Aspergillus fumigatus polymycovirus 1, Erysiphe necator associated deltaflexivirus 1, Alternaria alternata polymycovirus 1, Ingleside virus. All these possible due to Machine Learning techniques such as Hierarchical clustering method. These Methods or algorithms can be further improved to solve genomics and proteomics problems.

VI. REFERENCES

[1]. Venkataraman, S., Prasad, B., & Selvarajan, R. (2018). RNA Dependent RNA Polymerases: Insights from Structure, Function and Evolution. Viruses, 10(2), 76. https://doi.org/10.3390/v10020076
[2]. [Internet]. https://www.displayr.com/what-is-hierarchical-clustering/
[3]. Nielsen, Frank (2016). "8. Hierarchical Clustering". *Introduction to HPC with MPI for Data Science*. Springer. pp. 195–211. ISBN 978-3-319-21903-5.
[4]. Protein [Internet]. Bethesda (MD): National Library of Medicine (US), National Center for Biotechnology Information; 2021. Available from: https://www.ncbi.nlm.nih.gov/protein
[5]. Fabian Pedregosa, Gaël Varoquaux, Alexandre Gramfort, Vincent Michel, Bertrand Thirion, Olivier Grisel, Mathieu Blondel, Peter Prettenhofer, Ron Weiss, Vincent Dubourg, Jake Vanderplas, Alexandre Passos, David Cournapeau, Matthieu Brucher, Matthieu Perrot, Édouard Duchesnay. **Scikit-learn: Machine Learning in Python**, Journal of Machine Learning Research, **12**, 2825-2830 (2011
[6]. Pauli Virtanen, Ralf Gommers, Travis E. Oliphant, Matt Haberland, Tyler Reddy, David Cournapeau, Evgeni Burovski, Pearu Peterson, Warren Weckesser, Jonathan Bright, Stéfan J. van der Walt, Matthew Brett, Joshua Wilson, K. Jarrod Millman, Nikolay Mayorov, Andrew R. J. Nelson, Eric Jones, Robert Kern, Eric Larson, CJ Carey, İlhan Polat, Yu Feng, Eric W. Moore, Jake VanderPlas, Denis Laxalde, Josef Perktold, Robert Cimrman, Ian Henriksen, E.A. Quintero, Charles R Harris, Anne M. Archibald, Antônio H. Ribeiro, Fabian Pedregosa, Paul van Mulbregt, and SciPy 1.0 Contributors. (2020) **SciPy 1.0: Fundamental Algorithms for Scientific Computing in Python**. *Nature Methods*, 17(3), 261-272.
[7]. Chapman BA and Chang JT (2000). Biopython: Python tools for computational biology. ACM SIGBIO Newsletter, 20, 15-19
[8]. [Internet]. https://biopython.org/docs/1.75/api/Bio.Entrez.html
[9]. [Internet]. https://www.kaggle.com/
[10]. Papadopoulos JS and Agarwala R (2007) COBALT: constraint-based alignment tool for multiple protein sequences, Bioinformatics 23:1073-79. PubMed.
[11]. [Internet]. https://www.ncbi.nlm.nih.gov/tools/cobalt/cobalt.cgi?

VII. ACKNOWLEDGMENT

This paper and the research behind it would not have been possible without Python, Biopython, NCBI and Friend's support.