Influenza Virus Sequence Prediction using Neural Networks

Team Number : 4
Team Name : Go Corona

Contributors

Anmol Kumar
Samarth Chauhan
Ritvik Gupta
Karan Chawla
Anjali Sharma

Abstract

Influenza remains to be one of the biggest viral infections and has a huge impact on global health every year. On an average, around 50,000 people die every year due to flu infection. However, if one is vaccinated, the risk of getting this infection is minimal. With the constant evolution of the viral strains and the mortality and morbidity rate of the viruses, it becomes highly necessary that we refine our methods for the effective drug production based medical research. In this project, we are going to forecast the influenza virus strain sequence that is likely to appear in the subsequent years based on the past data of the influenza virus in America.

Influenza Vs Covid 19: The speed of transmission is an important point of difference between the two viruses. Influenza has a shorter median incubation period (the time from infection to appearance of symptoms) and a shorter serial interval (the time between successive cases) than COVID-19 virus. The serial interval for COVID-19 virus is estimated to be 5-6 days, while for influenza virus, the serial interval is 3 days. This means that influenza can spread faster than COVID19. But the death rate for influenza is only 0.1% and the corona virus is around 5-6%.

Introduction

The influenza viruses in the seasonal flu vaccine are selected each year based on surveillance data indicating which viruses are circulating and forecasts about which viruses are the most likely to circulate during the coming season. The degree of similarity between available vaccine viruses and circulating viruses is also an important factor to consider. Vaccine viruses must be similar to the influenza viruses predicted to circulate most commonly during the upcoming season.

In this project, we are trying to keep track of the mutations in the Influenza A strain using time-series based machine learning methods. The model trains on the dataset of the past influenza viral strains and prediction of the future viral sequences is done using Time Series Forecasting. Deep Learning prediction methods like dense Neural Networks, ARIMA and LSTM(RNNs) are used for forecasting.

In order to enhance our forecasting model, positions which were SNP's were found using Multiple Sequence Alignment and only the SNP sites were replaced by the sequences predicted from the Time Series forecasting model.

Dataset Used:

We have taken the dataset for our project from FluDB as is the reliable site for virus strain or virus related data

(link: https://www.fludb.org/brc/influenza_sequence_search_segment_display.spg).

We have chosen the flu virus sequences from the dataset for the USA and India between 2000 and 2020.
Multiple sequences for the influenza virus are generated almost every year in the USA, so the highest amount of data was available for this country. This is the reason for choosing the USA over India or any other country.

So all the results we have shown here are for the USA data, the output for India or any other country can be generated using the appropriate dataset in our code.

Total number of Sequences Obtained from Dataset: 13552 for USA
552 for India

Search Parameter :

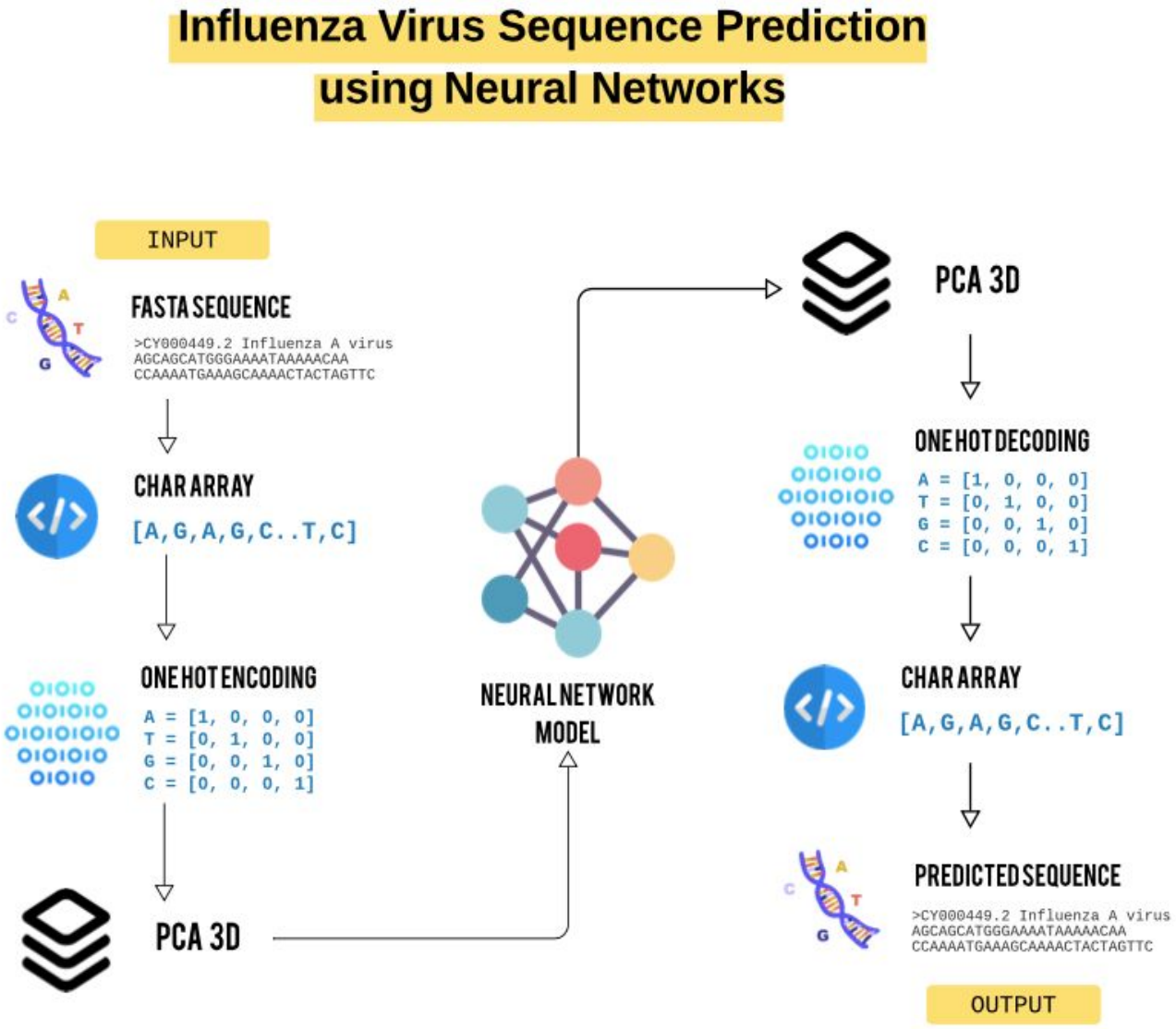
For USA

Data type: Protein
Virus Type: A
Subtype: H1N1
Date range: from 2000 to present
Classical protein: HA
Complete option: HA
Variant proteins (SOP): no option selected
Clade classification: None
Host: Human
Geographic grouping: North America
Country: USA
Advanced option: Remove duplicates sequence

For India

Data type: Protein
Virus Type: A
Subtype: H1N1
Date range: from 2000 to present
Classical protein: HA
Complete option: HA
Variant proteins (SOP): no option selected
Clade classification: None
Host: Human
Geographic grouping: Asia
Country: India
Advanced option: Remove duplicates sequence

Methodology:



Dataset Cleaning and Preprocessing :

1. Sequences were aligned using MAFFT from as it supports large file support and makes result fast as compared to other software or tools available online. For multiple alignment
2. Accession number extraction: Dataframe was created with accession number and sequence extracted from the fasta file.
3. Deleted data which doesn't contain any date.
4. Collection date in the metadata of sequence was not in the datetime format in which python can read so it was converted to datetime format and then in the format in which year is written first.
5. Sorting data on the basis of collections dates.
6. Checked if all data is for our purpose ie host, protein name, subtype of all sequence was checked again to ensure labelled data is processed.

Feature Preparation:

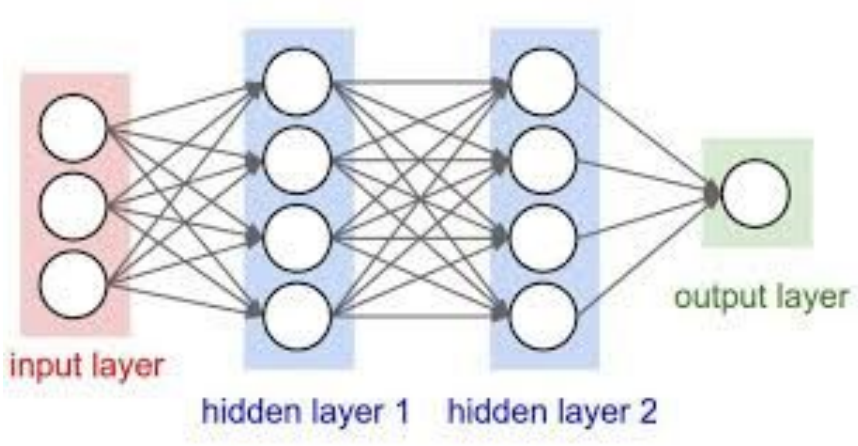
1. We checked which characters are present in the sequence and then a label binarizer was made which converted the character array into one hot encoding.
2. Sequence is converted to character array and then character array to 2D binary array so size will be (total number of sequence) x (unique characters in sequence * length of sequence).
3. Parameters were defined for splitting data into training and validation of data, like year, month, grouping parameter.
4. Variational autoencoder model was prepared using a dataset as defined in the above step then encoder and decoder is saved into memory to use it again.
5. Encoder and decoder models were loaded from memory.
6. Binary array of the whole data set was passed through the encoder to reduce dimension into 3, normally the sequence can be represented by a 12K size array but with the encoder model we can represent the same sequence with 3 dimensions only.
7. PCA or 3 dimension data is grouped using month, quarter or day, for example if 10 sequences are available for May 2019 month and we decided to group data by month then for May 2019 we take a median of 10 data. Now for every month we have one sequence.
8. Train and test defined above were passed to models.

Prediction Models and their Outputs along with Inferences:

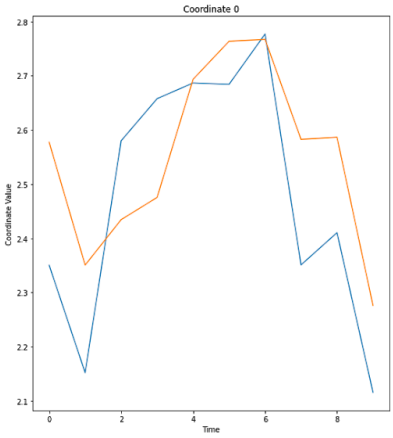
Input data to the model are time based windows of data. We have chosen our window length to be 5 which means we have taken a season length to be approximately 5 months.
For eg: based on months month 1-5 are taken in a batch and given in as input and the 6th month's value is given as the corresponding labels for training. The next set of training data is from month 2-6 and 7th month's labels are predicted and so on.

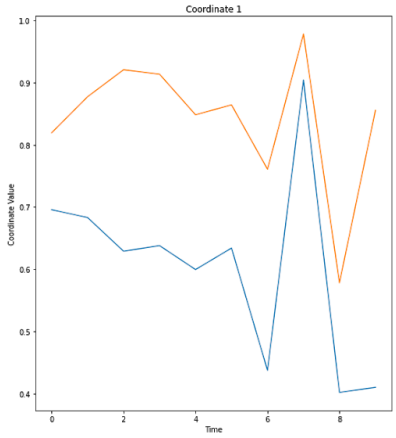
We have done prediction of data from 2019 to 2023 using 3 different prediction models.

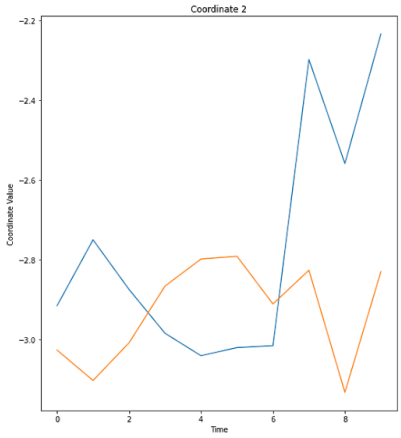
1) Dense Neural Networks:

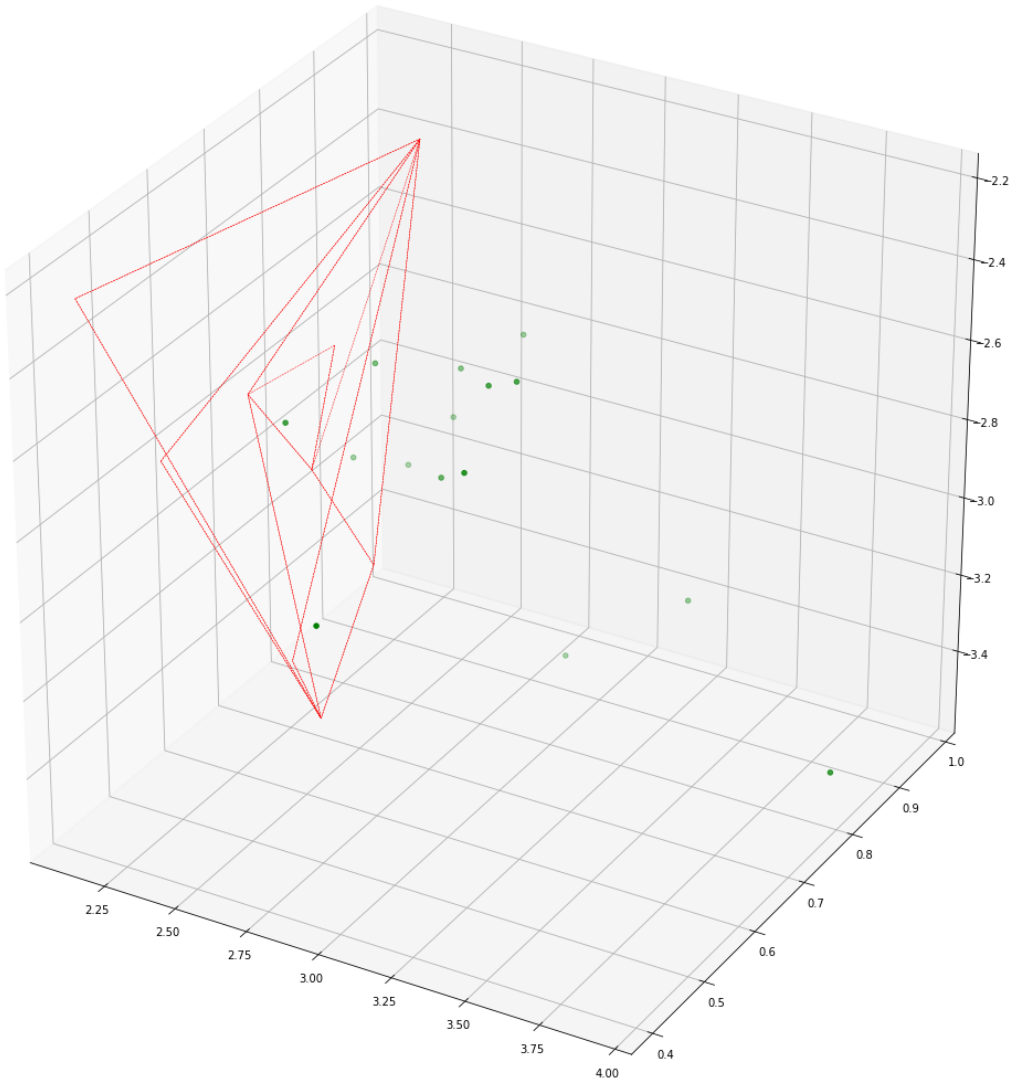


Before using the neural we have to process data once again so that it matches the condition of input size.
We have used a neural network with 2 hidden dense layers before the final output.
The 2 hidden dense layers have 10 units each and the output layer will have a single unit.
We have used the relu activation function which is the most commonly used in Neural networks.


Coordinate 0
Coordinate Value
Time

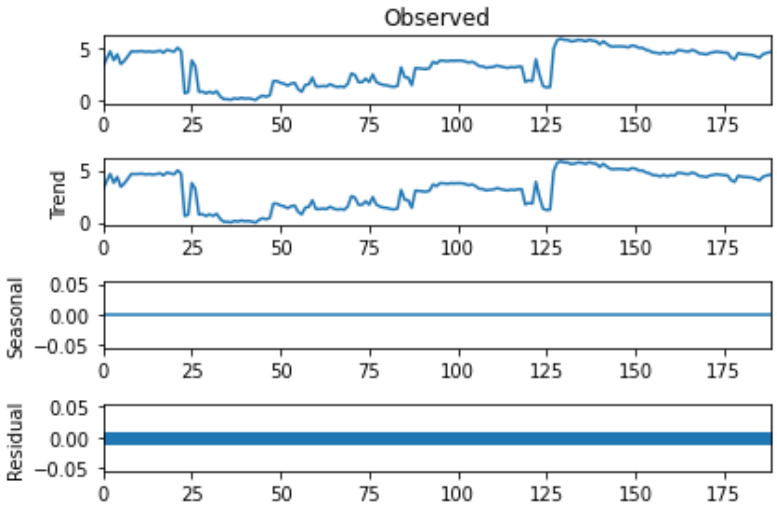

Coordinate 1
Coordinate Value
Time


Coordinate 2
Coordinate Value
Time


2.25
2.50
2.75
3.00
3.25
3.50
3.75
4.00
0.4
0.5
0.6
0.7
0.8
0.9
1.0
−2.2
−2.4
−2.6
−2.8
−3.0
−3.2
−3.4

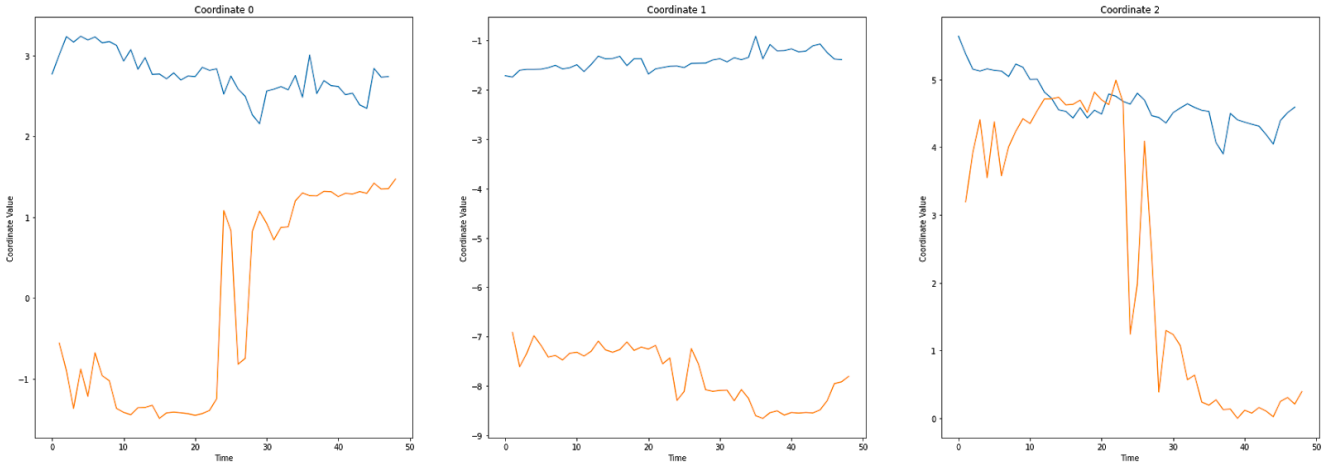
2) ARIMA

To get the best ARIMA model we used the auto_arima function and seasonal, trend graph on each dimension of PCA. For example seasonal decomposed graph of one dimension



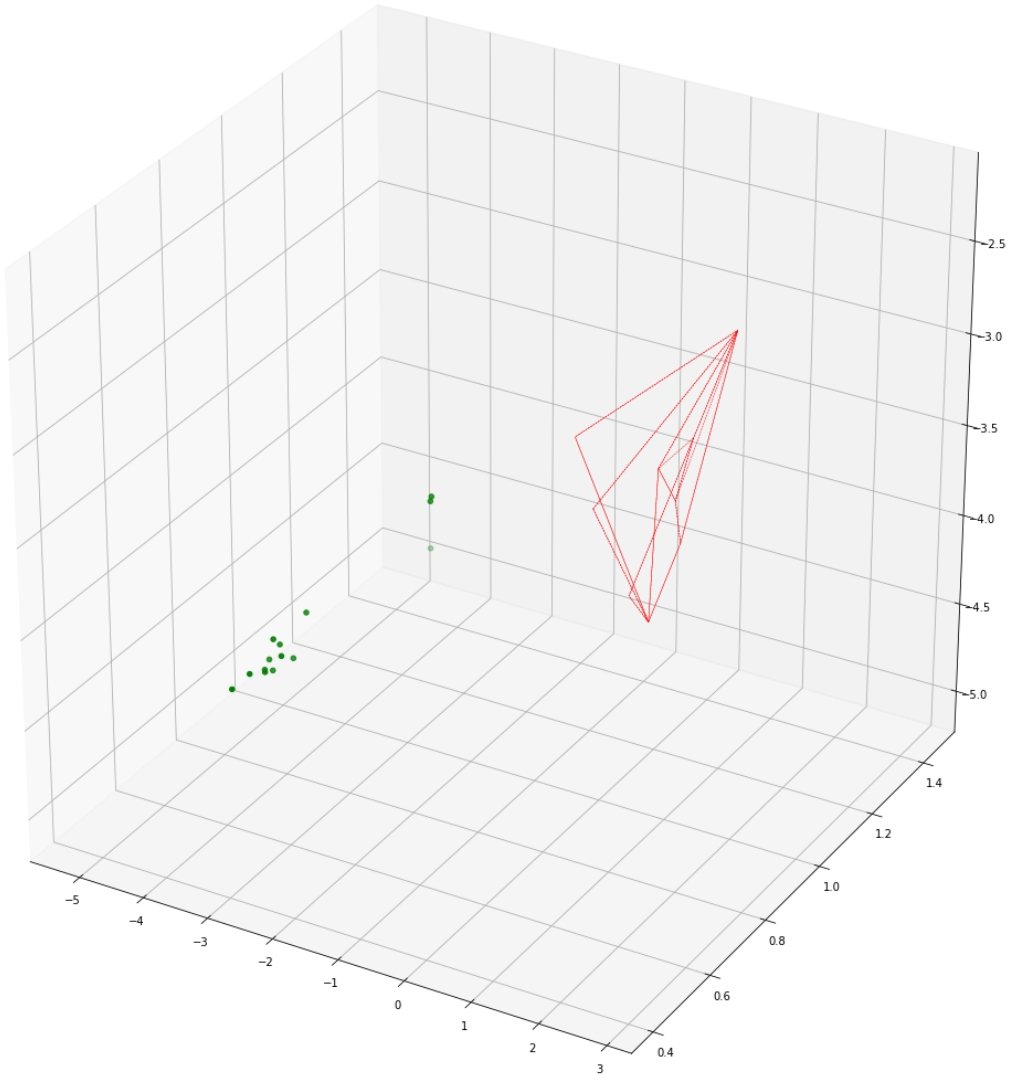
Then the best ARIMA model selected from the above process is trained on the PCA dimensions.

On our training and testing prediction of ARIMA on PCA were as followed



Blue line are the real data point on test data and,
Orange is the prediction of ARIMA model

In 3D space,



Box made by red line is the test data convex hull and,
Green dots are the prediction done be ARIMA model

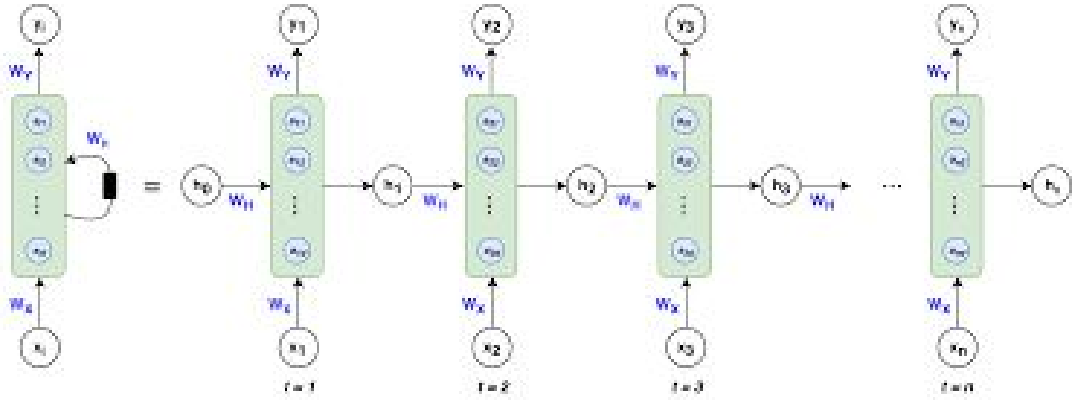
```
                              SARIMAX Results
==========================================================================================
Dep. Variable:                     coord0   No. Observations:                   65
Model:                 SARIMAX(0, 1, 0)   Log Likelihood                -95.911
Date:                  Wed, 20 May 2020   AIC                           193.822
Time:                          20:30:30   BIC                           195.980
Sample:                               0   HQIC                          194.672
                                    - 65
Covariance Type:                    opg
==============================================================================
                 coef    std err          z      P>|z|      [0.025      0.975]
------------------------------------------------------------------------------
sigma2         1.1727      0.130      9.017      0.000       0.918       1.428
===================================================================================
Ljung-Box (Q):                       45.30   Jarque-Bera (JB):                 28.66
Prob(Q):                              0.26   Prob(JB):                          0.00
Heteroskedasticity (H):               1.59   Skew:                              0.75
Prob(H) (two-sided):                  0.29   Kurtosis:                          5.92
===================================================================================

Warnings:
[1] Covariance matrix calculated using the outer product of gradients (complex-step).
                              SARIMAX Results
==========================================================================================
Dep. Variable:                     coord1   No. Observations:                   65
Model:                 SARIMAX(1, 0, 3)   Log Likelihood                -139.963
Date:                  Wed, 20 May 2020   AIC                           289.927
Time:                          20:30:30   BIC                           300.799
Sample:                               0   HQIC                          294.216
                                    - 65
Covariance Type:                    opg
==============================================================================
                 coef    std err          z      P>|z|      [0.025      0.975]
------------------------------------------------------------------------------
ar.L1          0.9264      0.090     10.267      0.000       0.750       1.103
ma.L1          0.1422      0.174      0.815      0.415      -0.200       0.484
ma.L2         -0.0869      0.118     -0.733      0.463      -0.319       0.145
ma.L3         -0.3468      0.144     -2.401      0.016      -0.630      -0.064
sigma2         4.2205      0.594      7.106      0.000       3.056       5.385
===================================================================================
Ljung-Box (Q):                       38.23   Jarque-Bera (JB):                 56.59
Prob(Q):                              0.55   Prob(JB):                          0.00
Heteroskedasticity (H):              10.05   Skew:                              1.15
```

3) LSTM with Optimization using Convolutional Neural Networks:

LSTM are a complex type of recurrent neural networks(RNNs). This model has the ability to memorise data over previous timestamps inputs and give feedback based on it.

To optimize the running time and efficiency of LSTM, we have added a layer of 1-d Convolutional Neural Network to it.
After that we have used 2 layers of LSTM of 16 units before getting the output in the final dense layer.

LSTM takes 3-D input and compulsorily generates a 3-D output.
It uses tanh activation function instead of RElu that we generally use in Neural Networks. This is because of its unstable gradient value. The output graph of LSTM has 5 predictions and their consensus value is taken for measuring accuracy.



Evaluation Criteria

Unlike normal Machine Learning and Deep Learning prediction models, the success of a time series model does not rely much on accuracy. Error value is always considered to be a better parameter.
Moreover mutations in a long virus sequence occur on a very few sites, so accuracy doesn't make sense in this case.

However we have measured both accuracy and mean absolute error.

Error in the Neural Network model:
2.1610112
2.6192698
2.5291216
Mean error : 2.436467488606771
This is a reasonably low error value

Error in the LSTM model:
42.1610112
44.6192698
41.291216
Mean error : 42.436467488606771
This is not low but also not a high error value.

Including SNPs in Predictions

Over the years, the influenza viral strain keeps on evolving and mutating and hence it becomes important to include SNPs in our neural network model. Due to this, our prediction's focus increases more on those sites. The results from these SNP predictions sites will be later included with our main time-series model. This gives us increased chances of better accuracy.

There are various methods to get the variant callings in the complete genomes of organisms. One of the techniques included using the GATK workflow pipeline consisting of bwa, bcftools and samtools. This method requires mapping each of the viral strain samples to the reference genome of influenza. But the problem with this method is that we can then only compare the sequences with the reference genome and get the variant callings for that time interval change. This would be unfruitful for us as the influenza viral genome sequences are generally sequenced during major outbreaks although the virus is constantly evolving. Also, there would be a lot of reference genomes to keep track of. The alignment of the sequences to the reference genomes and the sorting of .sam files for each sequence in the dataset will be a highly time and space consuming task as well.

Thus, we have used an efficient work-around by using the **MAFFT** multiple sequence alignment tool. Alignment using MAFFT is time and space efficient and it gives almost the same results as we would have by using GATK. The alignment is downloaded in the form of FASTA format and it is fed as input to the **snp-sites** tool which gives us information about all the SNPs including their positions and sequences in the form of .vcf and .aln format respectively. We can then use a small script (included in the notebook itself) to get all the snp-positions in the form of a numpy array and use that in our neural networks further.

Further Improvements:

Although we were able to keep the error level at a very low value, our models, especially LSTM and ARIMA might not be giving the best possible prediction of the virus sequence data.
We think so because of the following reasons:

1) Non availability of a good dataset with equal time-stamped values based on a long period of time: Although the USA dataset had a very large number of sequences very few were of our use. Only 200 out of 13000 were left after cleaning.
2) Not enough knowledge about the underlying Math for models like ARIMA and LSTM, which otherwise would have helped us in setting the large number of parameters in these complex models.
3) We considered only sequences for making feature matrices. Several other temporal parameters like population, life expectancy at that time etc could have been considered for making the model.

4) The data is generalized for the USA i.e. we have not considered the data from a particular city instead we took all the data available.

Conclusion:

Although there is no particular way to predict accuracy per se in time series forecasting(No one can predict the future), based on our model's outputs and predictions we can say that our model does well to predict future influenza sequences. This project can be further improved and used for developing various tools for analysis of influenza virus and can help in predicting the vaccine for the next influenza virus in advance by mutation prediction. It can also facilitate flu surveillance and help in the study of evolution of influenza sequences.
Sequence generated from all above model is in txt form with model name for eg ARIMA.txt
In all the result files, these contain sequences from the model which are 48 months in advance with respect to the train data that is 2019 january. Data from 2019 to 2020 march is used for validation and remains as the prediction.

References:

www.nature.com/articles/nrd4529
https://content.iospress.com/articles/in-silico-biology/isb00345
https://www.ncbi.nlm.nih.gov/pmc/articles/PMC5861780

Databases and relevant websites:
https://www.fludb.org/brc/vaccineRecommend.spg?decorator=influenza
https://www.cdc.gov/flu/weekly/pastreports.htm https://www.ncbi.nlm.nih.gov/gene
https://mafft.cbrc.jp/alignment/software/