A BiLSTM Based on Virus Sequence Data

DD2402 Advanced Individual Course in Computational Biology

Sabrina Chowdhury

May 2021

Abstract

Understanding the language of viral mutations as well as how they occur has become a very relevant topic with the recent COVID-19 pandemic that has reached almost all corners of the world. A recent study conducted by researchers at MIT showed that applying machine learning models traditionally used for human language on viral sequence data can be effective in predicting viral escape. This research project aims at investigating and recreating a subset of the experiments in the study concerning the accuracy of the network as well as the cluster purity of the semantic embeddings learned by the network. The main results show that even with a significantly smaller dataset, the network accuracy is very high given that the distribution with respect to host species in both training and validation sets are approximately the same. The results further show that the network fails to cluster the sequences according to host species.

Contents

1 Introduction

With the onset of recent pandemics, especially the ongoing COVID-19 pandemic, more and more people around the world have come to understand the importance of research that can aid in preventing new viruses [3]. With new viruses also comes the risk of new viral mutations that can cause additional waves of infection even after the virality of the original strand has been lowered. This research project therefore aims at investigating the performance of a BiLSTM network on Influenza HA viral sequence data as well as how well the network clusters viral sequences according to host species. These experiments are meant to be a slightly modified reproduction of the experiments carried out in the Viral Escape study [1], which showed that language models could be used to learn viral escape patterns. Although this study does not directly investigate whether the network can accurately predict viral escape, it does investigate the accuracy of the network as well as its internal representation of the viral sequences it is trained on. This can potentially aid in providing more thorough evidence for the effectiveness of this method as well as its possible downfalls.

1.1 Purpose

This research project aims at attempting to replicate a subset of the experiments in the Viral Escape study [1]. This research project first aims at investigating whether the BiLSTM network can achieve good training and validation results when trained on only 1,500 Influenza HA viral sequences as opposed to the 45,000 used in the Viral Escape paper. Furthermore it investigates the significance of training and validating sets having different host species distributions. The study also aims at attempting to replicate the UMAP and Louvain clustering results which showed meaningful UMAP groupings as well as high cluster purities in terms of host species.

1.2 Research Question

The research question this study aims at answering can be broken down into the following three questions :

1. To what extent can a BiLSTM trained on only 1500 viral sequences, as opposed to 45000 viral sequences, achieve correct classification results - and how significant is the host species distribution in training and validation sets to its classification results?

2. How pure are the clusters of semantic embeddings generated by the trained BiLSTM with respect to host species?

3. How important is the order of input to the KNN algorithm to the appearance of the UMAP clustering as well as the Louvain clustering purity?

1.3 Scope

The experiments done in this project have been altered to an extent compared to the experiments performed in the study conducted by Hie et al. [1]. Firstly, instead of experimenting on three different proteins, due to time constraints, only HA Influenza viral sequences are used in project. Secondly, instead of focusing on both subtype and host species as metadata categories, this project is solely focused on the host species category. Thirdly, instead of using 45,000 sequences to train the network, 1,500 sequences are used. Lastly, instead of using a BiLSTM architecture with 512 nodes in each BiLSTM layer the network used in this project has 100 nodes.

2 Background

2.1 Viral Escape

Learning the Language of Viral Evolution and Escape is a research study [1] conducted by researchers at MIT where the main aim is to model viral escape with machine learning algorithms originally developed for human natural language. The researchers describe viral escape as the ability of viruses to mutate and evade the human immune system to cause infection. This is an obstacle to antiviral and vaccine development, which could be aided by a greater understanding of how viral escape occurs. To escape a mutant virus must preserve infectivity and evolutionary fitness, meaning that it must obey some biological "grammar" while avoiding being detected as a threat by the immune system. The MIT researchers see this as analogous to a change in the meaning or semantics of the virus, much like changing the meaning of a sentence by switching out a word. The researchers trained a language model on news headlines where the network was trained to predict the missing word in a headline given the rest of the headline. The network would in this way learn the probability of a given word given the context, something the researchers call "grammaticality" as it is a representation of how well the word fits in the context of the rest of sentence. Internally the network also constructs a semantic embedding space through its weights. The researchers use the term semantic change to describe the difference between the semantic embedding of a mutated sequence and the semantic embedding of the original sequence. A smaller semantic change is then analogous to the immune system being less able to see a difference between the mutated sequence and the original sequence, meaning that the mutated sequence will more easily evade being detected as a threat by the immune system. The researchers use semantic change together with grammaticality to determine how likely viral escape is for a given mutated viral sequence. The researchers did a grid search with 36 different models and parameters to ultimately arrive at a BiLSTM architecture which produced the best results. The researchers also visualized the semantic landscape learned by the model using the dimensionality reduction method UMAP, which resulted in a 2D semantic landscape. The study found that the UMAP landscape showed clusters corresponding to the subtype, host

species, or both which suggested that the model was able to learn functionally meaningful patterns from the viral sequences. The researchers also used Louvain community detection to cluster the semantic embeddings and found a high cluster purity with respect to both subtype and host species.

2.2 Recurrent Neural Networks (RNNs)

A recurrent neural network (RNN) [4] is used when either the input or output data of a neural network is sequential. This could e.g. be words in a sentence or images of the same object during different days. RNNs are used in e.g. human language translation (many-to-many model), predicting some word in a sentence given the preceding words (many-to-one model) or image captioning (one-to-many model). Building a neural network to fit this type of task where sequential data is involved requires the addition of a temporal dimension.

To describe the structure of an RNN model one can start with the simplest building block of neural networks, the perceptron, which processes a weighted input vector w*x which goes through a nonlinear activation function to generate an output y. A feedforward neural network is simply one or several layers consisting of perceptrons. To be able to accommodate for sequential data the feedforward network has to be able to take several time steps into account. One instance of the feedforward network can be seen as one time step. To incorporate several time steps, several instances of the feedforward network is created (one for each time step, also called an NN cell). However, when treating the NN cells as isolated time steps one loses information that can be shared from previous time steps, and so memory is introduced which is passed from the previous cell to the next (also called the cell state). This addition of memory means that the output of a given cell depends both on the input for that time step as well as the previous cell state (memory) which was passed to the current cell from the previous cell. This is what makes each cell a recurrent cell (RNN cell). The new cell state for the current time step then depends on both the previous cell state, the current input as well as the three weight matrices that are updated during training (one for going from the previous to current cell state, one for weighting the input to the cell and one going for transforming the current hidden cell state to the output).

In backpropagation the loss is then backpropagated through the different time steps and is associated with the vanishing gradient problem as the information that is backpropagated can easily become smaller and smaller with the number of time steps (due to the multiplication of a growing number of small gradients together through the time steps). Thus it becomes increasingly difficult to backpropagate errors and the weights become biased towards capturing short-term dependencies. A recent and robust solution to this is the LSTM which uses a gated cell.

2.3 Long Short Term Memory Network (LSTM)

The LSTM [2] introduces a type of cell which controls the flow of information with the help of gates. Information is thus either added or removed through the different types of gates. The LSTM works by first forgetting irrelevant past information, storing the most relevant new information, updating the current internal cell state and finally generating an output. The forgetting is done by taking the previous cell state and passing it through a sigmoid gate, where a sigmoid activation function forces everything that passes through it to only take values between 0 and 1. In this way the cell can modulate how much information should be let in. The LSTM overcomes the vanishing gradient problem by allowing for uninterrupted gradient flow, through a separately kept cell state where the backpropagation takes place.

2.4 Bidirectional Long Short Term Memory Network (BiLSTM)

Using an LSTM is very suitable in the context of viral sequences as the LSTM network can be trained to predict the last amino acid in a sequence given the sequence up until the penultimate amino acid. Mutations can, however, occur anywhere in the sequence and thus it is of interest that the LSTM can be trained to predict a missing amino acid in a sequence given the rest of the sequence. This requires information about both the preceding sequence as well as the succeeding sequence to the amino acid of interest. This is where the bidirectional LSTM [5] comes in, where each LSTM cell passes information in both the forward and backward direction such that a given cell can access information about both the preceding sequence as well as the succeeding sequence. The BiLSTM is slower than the LSTM, but is more suitable for the type of data used in this experiment, as was also shown in the Viral Escape paper.

3 Method

3.1 Data collection

HA Influenza sequences are collected from fludb.org with the following metadata categories : Data type (protein), Virus type (A), Host (Avian, Swine, Human), Complete? (HA). The host species distribution on the website (on the 7th of April 2021) can be found in Appendix A. The chosen dataset consisted of the 3 most frequently reported host species (above a set threshold of 10,000 sequences) : Human, Avian and Swine. To acquire approximately 10,000 sequences from each of the three host species categories a certain date range is used for each host species category and can be found in Table 1.

Host Species	Number of Sequences	Date Range
Human	9,897	2018 (feb) -2019(dec)
Avian	9,980	2014 (may) -2019(dec)
Swine	9,852	2012 - 2019

Table 1: Date ranges for each host species category

From this acquired dataset of approximately 30 000 viral sequences, 500 unique viral sequences from all three host groups are chosen to make a total of 1500 unique viral sequences in the dataset. The BiLSTM network is then trained on a total of 840000 mutated sequences (how these sequences are acquired is described in the next section).

3.2 Preprocessing of data

All inputs to the BiLSTM have to be equal in length and therefore each viral sequence in the dataset has to first be padded out to a length equal to the longest viral sequence in the dataset. In this study each viral sequences is left-padded with the character "*". The longest sequence in the dataset used was 568 and therefore all masked sequences were padded to be of length 568.

The aim of training the BiLSTM model is for it to be able to correctly predict some amino acid when given a viral sequence with that particular amino acid masked out. The next step in preprocessing is therefore to create a number of new sequences (henceforth called mutated sequences) with one amino acid masked out with the character "_". If a certain viral sequence has N amino acids, then in this stage of preprocessing N new mutated sequences are created. Finally 840 000 mutated sequences are created for training the BiLSTM.

Each viral sequence consists of a sequence of letters representing the different amino acids they contain. The neural network can only work with numerical data and therefore the next step of preprocessing is to tokenize each sequence according to the conversion table found in Appendix B.

Finally one-hot encoded target vectors describing the correct amino acid for each mutated sequence is created. Each array is of size 26 as there are 26 possible amino acids in the dataset.

A visual diagram for all the steps described in this section can be seen in Figure 1.

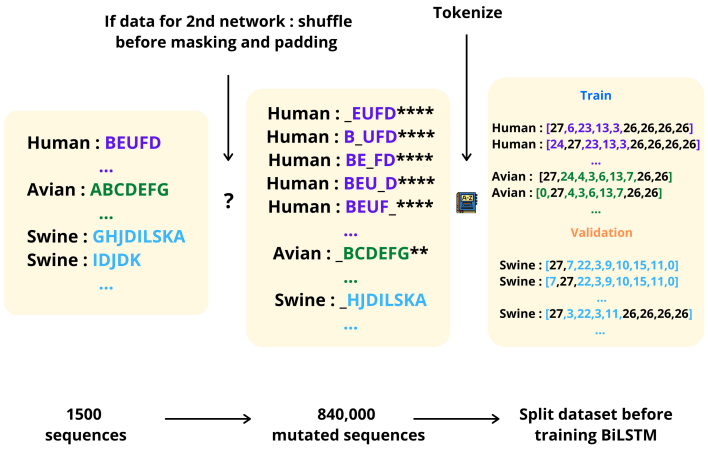

Figure 1: Visual diagram for how to preprocess the data

3.3 The model

The neural network model used in this study is a BiLSTM network with the following hyperparameters : an embedding layer after the input layer with a vector size of 20, 2 bidirectional layers with 100 nodes each (as opposed to 512 in the original Viral Escape study, due to computational reasons), an Adam optimizer with a learning rate of 0.001, beta1 = 0.9, beta2 = 0.999, a categorical cross entropy loss function. No batch size was mentioned in the Viral Escape paper, however, this study uses a batch size of 32. A visual diagram of the model with example inputs and outputs can be seen in Figure 2.

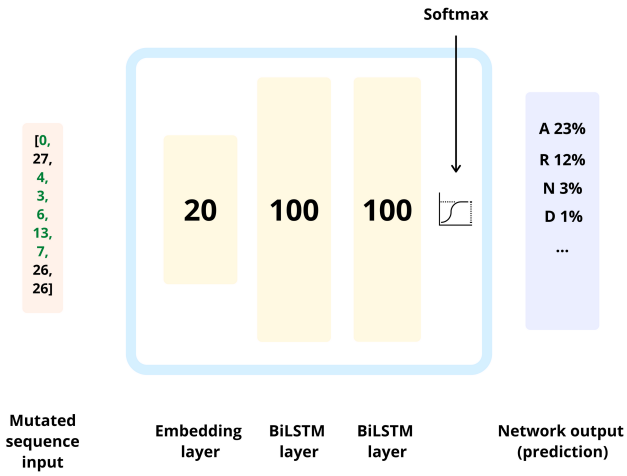

Figure 2: Visual diagram for the BiLSTM model

3.4 Experiments

3.4.1 Ordering vs Shuffling training and validation data according to host species before training BiLSTM

In the first run the network is trained and validated on separate host species. The network is trained on Human and Avian single mutated Influenza sequences meanwhile it is validated on Swine single mutated Influenza sequences. The phenomenon of interest here is to investigate how significant a similar host species distribution in both training and validation sets is to network performance. This is related to understanding how varied the sample size has to in terms of host species in order for the network to understand the underlying distribution for all Influenza viral sequences.

In the second run the network is trained and validated on all three host species. The dataset is shuffled before training such that both training and validation sets have the same distribution of host species.

3.4.2 Ordering vs Shuffling KNN input data according to host species

In the next experiment it is of interest to investigate both qualitatively and quantitatively how the network organizes the original 1500 viral sequences in semantic embedding space. The network used in this test was the one trained and validated on shuffled data (the second run in the previous section). Here it is of interest to see if it makes a difference whether or not the input sequences is shuffled (with regards to host species) before generating a neighborhood graph with the KNN algorithm. This KNN neighorhood graph is then sent into UMAP

for dimensionality reduction (2 and 3 output dimensions) which is qualitatively assessed in plots. The KNN neighborhood graph is also sent into the Louvain clustering method for quantitive assessment of clustering purity. A visual diagram for the steps described in this section can be seen in Figure 3.

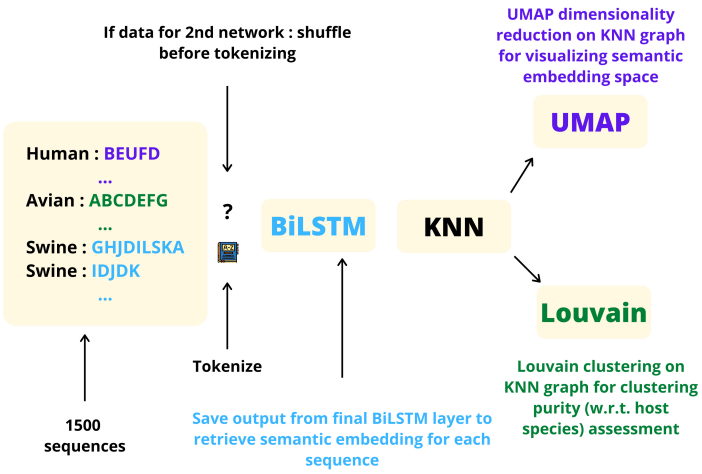

Figure 3: Visual diagram for KNN input order experiment

4 Results

4.1 Shuffled training and validation data according to host species before training BiLSTM

As can be seen in Figure 4 and Figure 5 the network reaches a high final accuracy and a low final loss without displaying overfitting behaviour when trained and validated on data with similar distributions of host species.

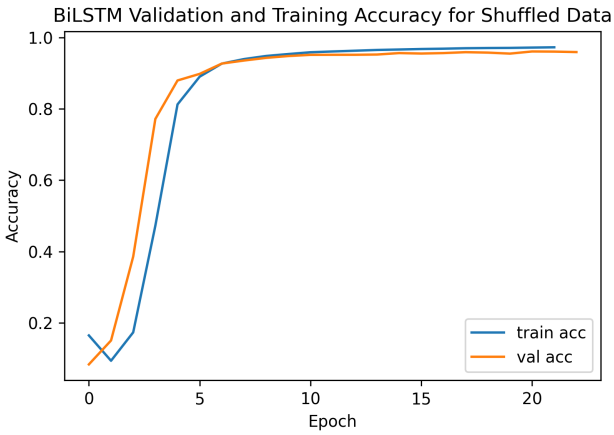

Figure 4: Validation and training accuracy for shuffled data

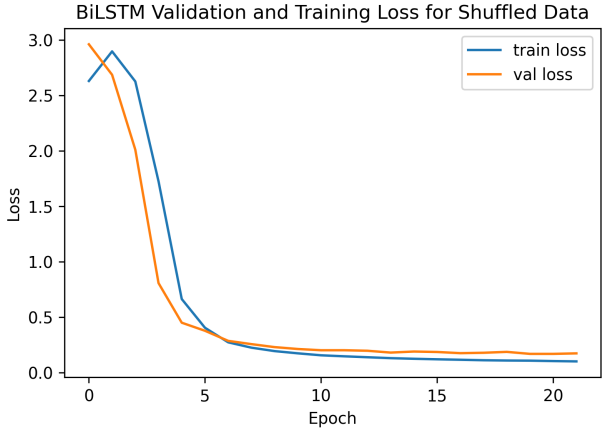

Figure 5: Validation and training loss for shuffled data

4.2 Ordered training and validation data according to host species before training BiLSTM

As can be seen in Figure 6 and Figure 7 the network reaches a relatively low final validation accuracy and a relatively high final validation loss in comparison to the results achieved by the network trained on shuffled data in the previous section. This network also displays overfitting behaviour when trained and validated on data with dissimilar distributions of host species (trained on Human

and Avian sequences while validated on Swine sequences).

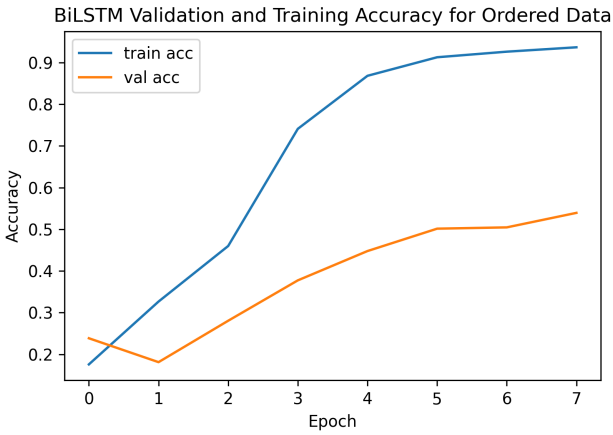

Figure 6: Validation and training accuracy for ordered data

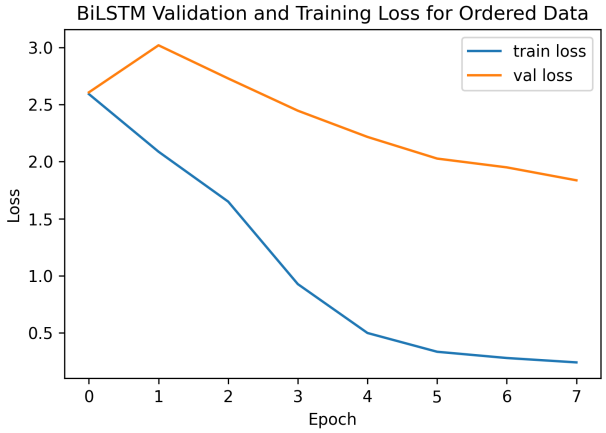

Figure 7: Validation and training loss for ordered data

4.2.1 Ordered KNN input data according to host species

As can be seen in Figure 8 and Figure 9 ordering the KNN input data according to host species results in what qualitatively can be assessed as a grouping with meaningful clusters. The 3D plot seems to display a ring shaped manifold meanwhile both plots display relatively clear groupings or clusters with respect

to host species.

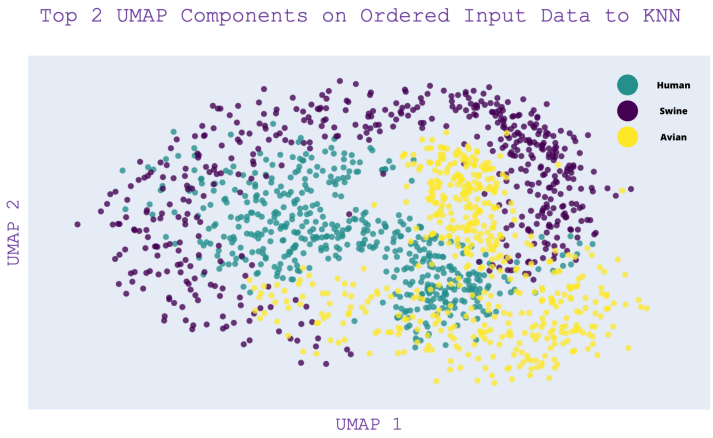

Figure 8: 2D UMAP plot for ordered KNN input data

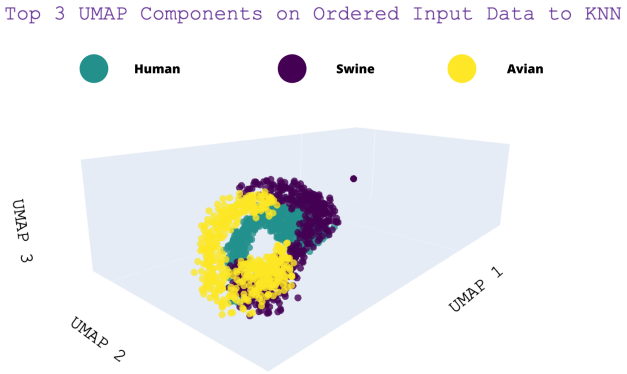

Figure 9: 3D UMAP plot for ordered KNN input data

4.2.2 Shuffled KNN input data according to host species

As can be seen in Figure 10 and Figure 11 shuffling the KNN input data according to host species results in what qualitatively can be assessed as a grouping without meaningful clusters. The 3D plot, much like the previous section with ordered input data, seems to display a ring shaped manifold however the distribution of host species is no longer assembled into any coherent groupings in any of the two plots.

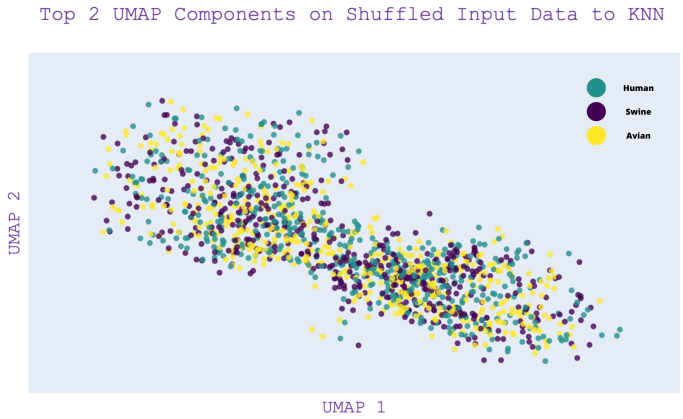

Figure 10: 2D UMAP plot for shuffled KNN input data

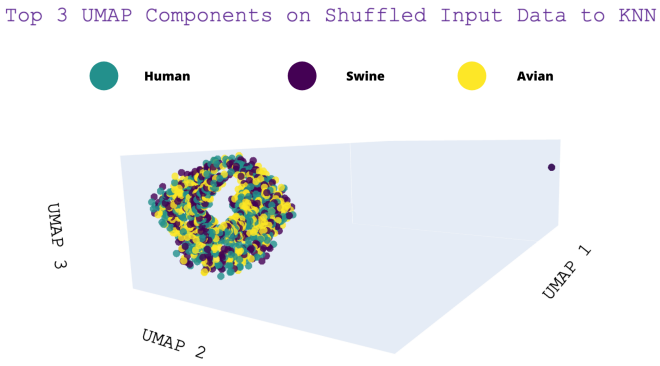

Figure 11: 3D UMAP plot for shuffled KNN input data

4.2.3 Louvain clustering purity

Louvain community detection clustered the 1500 sequences into two clusters with 1489 sequences in the first cluster, meanwhile only 11 sequences were assigned to the second cluster. The results were identical for both ordered and shuffled input to the KNN algorithm.

5 Discussion

5.1 Discussion of results

The results suggest that the network performs significantly better when trained on data where the training and validation sets have equal distributions in terms of host species. When trained on Human and Avian sequences while validated on Swine sequences the network quite quickly starts to overfit as can be seen in 6 and Figure 7. This in comparison to when the BiLSTM network is trained on shuffled training and validation data where it displays a high final accuracy and low final loss as can be seen in Figure 4 and Figure 5.

In terms of the UMAP results it can be deduced that the input order to the KNN algorithm makes quite a significant difference in terms of appearence of the UMAP plots. Ordered input data generates a 2D and 3D graph with relatively meaningful regions in terms of host species in comparison to when the input data is shuffled and the plots display more random host species clusterings. This might suggest that the the KNN algorithm is more sensitive to the data input order as it seems to play quite a significant role in how the neighborhood is constructed.

Finally the Lovain clustering purity is the same regardless of whether the input to the KNN algorithm is ordered or not. An optimal clustering result would be 3 clusters with 500 sequences, all from the same host species, in each. The results in this study, however, show that almost all 1500 semantic embeddings of the viral sequences are clustered into one single cluster. A possible critique of this is perhaps that the algorithm clustered in terms of some other metadata category other than host species which this study has not taken into account, e.g. the subtype metadata category.

5.2 Possible future improvements

One interpretation of the results in this study is that the BiLSTM network used in this study needed a similar distribution in terms of the host species in both training and validation sets to avoid overfitting behaviour. It would therefore be interesting to see if this same behaviour could be replicated using other combinations of host species or even other metadata categories. The Viral Escape paper noticed that their network could see meaningful patterns in terms of both host species and subtype and thus subtype would be one example of a suitable metadata category to investigate further.

Another possible area of investigation could be to study which particular host species is more likely to spread a particular disease to some other species. In this particular study the results suggest that a network trained on Human and Avian sequences performs relatively poorly on Swine sequences. One interesting continuation of this experiment would be to e.g. train on Swine and Human sequences, and then compare its validation performance on Avian sequences with the performance presented in this study. These types of comparisons might give a greater insight into which species have more similar sequences for a given

virus.

It would further also be interesting to redo the experiment with other types of viruses other than the HA Influenza sequences used in this study.

6 Conclusion

The first part of the research question *"To what extent can a BiLSTM trained on only 1500 viral sequences, as opposed to 45000 viral sequences, achieve correct classification results - and how significant is the host species distribution in training and validation sets to its classification results?"* is answered by observing that the accuracy of the BiLSTM reaches an accuracy above 95 percent when trained and validated on sets with the same host species distribution.

The second part of the research question *"How pure are the clusters of semantic embeddings generated by the trained BiLSTM with respect to host species?"* is answered by observing that almost all viral sequences were clustered into one single cluster although there being three separate host species categories in the dataset. This is very dissimilar to the result observed in the Viral Escape paper which observed high cluster purities with respect to both host species and subtype.

The third part of the research question *"How important is the order of input to the KNN algorithm to the appearance of the UMAP clustering as well as the Louvain clustering purity?"* is answered by observing that the order of input according to host species significantly changes the appearance of the UMAP plots, but has no effect on the clustering purity.

Possible future improvements to get more definitive answers to this query include taking into account other metadata categories other than host species, as well as redoing the experiment with other types of viruses.

References

[1] Brian Hie et al. "Learning the language of viral evolution and escape". In: *Science* 371.6526 (2021), pp. 284–288.

[2] Sepp Hochreiter and Jürgen Schmidhuber. "Long short-term memory". In: *Neural computation* 9.8 (1997), pp. 1735–1780.

[3] World Health Organization et al. "Coronavirus disease (COVID-19)". In: (2020).

[4] David E Rumelhart, Geoffrey E Hinton, and Ronald J Williams. *Learning internal representations by error propagation*. Tech. rep. California Univ San Diego La Jolla Inst for Cognitive Science, 1985.

[5] Mike Schuster and Kuldip K Paliwal. "Bidirectional recurrent neural networks". In: *IEEE transactions on Signal Processing* 45.11 (1997), pp. 2673–2681.

A

Fludb.org Host Species Distribution on the 7th of April 2021

Host Species	Quantity
Human	51800
Avian	28296
Swine	13157
Dog	284
Equine	241
Domestic Cat	31
Horse	24
Pika	13
Crane	12
Large Cat	12
Sea Mammal	12
Ferret	11
Donkey	7
Mink	6
Fowl	5
Bat	4
Camel	2
Weasel	2
Anteater	1
Civet	1
Insect	1
Lion	1
Meerkat	1
Muskrat	1
Panda	1
Rat	1
Skunk	1

B

Conversion table for tokenization of amino acids and masking/padding characters

Sequence character	Integer code
'A'	0
'R'	1
'N'	2
'D'	3
'C'	4
'Q'	5
'E'	6
'G'	7
'H'	8
'I'	9
'L'	10
'K'	11
'M'	12
'F'	13
'P'	14
'S'	15
'T'	16
'W'	17
'Y'	18
'V'	19
'X'	20
'Z'	21
'J'	22
'U'	23
'B'	24
'Z'	25
'*' - for padding	26
'_' - for masking	27