Artificial neural networks and rough set theory in the prediction of zoonotic mutations in viral haemorrhagic fevers.

Raymond Kiganda

MSc in Data Science
The University of Bath
April 2020

Artificial neural networks and rough set theory in the prediction of zoonotic mutations in viral haemorrhagic fevers.

Artificial neural networks in the prediction of mutations in viral haemorrhagic fevers that enable the viruses to become zoonotic.

Submitted by: Raymond Kiganda

COPYRIGHT

Declaration

This dissertation is submitted to the University of Bath in accordance with the requirements of the degree of MSc Data Science in the Department of Computer Science. No portion of the work in this dissertation has been submitted in support of an application for any other degree or qualification of this or any other university or institution of learning. Except where specifically acknowledged, it is the work of the author.

Signed:

Artificial neural networks and rough set theory in the prediction of zoonotic mutations in viral haemorrhagic fevers.

Abstract

The aim of this project is to create a machine learning model that can accurately and reliably predict viral mutations for viruses that may become zoonotic. Using a feedforward neural network, rough set genetic mutation algorithm and the OpenNMT library, the project demonstrates that predicting viral mutations accurately is currently possible. It then goes on to suggest that advances in natural language processing and recurrent neural networks could prove useful in predicting genetic mutations.

Artificial neural networks and rough set theory in the prediction of zoonotic mutations in viral haemorrhagic fevers.

Contents

List of Figures

List of Tables

Acknowledgements

Would like to thank the Uganda Virus Research Institute for the invaluable viral data. Would also like to thank my supervisor Prof. Olga Isupova who was very understanding and helpful in, what has been, a very chaotic time. Would also like to thank Prof. Tom Haines for his help with accessing the computing power on campus. Finally, would like to thank the Bath Computer Science department.

Chapter 1

1 Introduction

Understanding and treating pathogens has been an area of research since they were discovered and the more that was understood about them, the greater the challenge of treating them became. This project will focus on viral pathogens, originally first discovered in the 1890s by Martinus Beijernick and Dimitri Ivanovsky during their work on a plant pathogen called *Tobamovirus* (Tobacco mosaic virus) (Beijerinck, M.W., 1898). A zoonotic virus is described as a virus which can infect both animals and humans and some viruses are able to develop this ability through genetic mutation (Zoonoses, 2020).

Viral particles are smaller than bacteria and could not be filtered and imaged until 50 years after Beijernick and Ivanovsky's original discovery when ultrafine filters were available to isolate *Poliovirus*, and the electron microscope was able to image them (Goldsmith and Miller, 2009). Since that time there has been no cure found for viral pathogens, only immune modulators (Le Calvez, Yu and Fang, 2004).

Viruses have also been responsible for the some of the deadliest pandemics in recent memory, Spanish Influenza (flu) (Saunders-Hastings and Krewski, 2016), Marburg (Stawicki et al., 2014), Severe Acute Respiratory Syndrome (SARS) (Zhong and Zeng, 2006), Ebola (Stawicki et al., 2014), Human Immunodeficiency virus (HIV) (Sharp and Hahn, 2011), and most recently COVID-19 (corona virus). The most effective way to treat a virus is using vaccinations, in which a weakened form of the virus is infected into the body to invoke an immune response and once the immune system has recognised the virus it can effectively handle any future infections with relative ease (Le Calvez, Yu and Fang, 2004). However, viruses have an incredibly high rate of mutation and if these mutations affect the identification proteins on the virus the immune system will no longer recognise them and will be unable to combat the virus effectively (Sanjuán and Domingo-Calap, 2016). The seasonal change of the influenza vaccine is a good example of this (Houser and Subbarao, 2015).

As with all pathogens some viruses can infect multiple species whilst others are limited to one. The high rate of mutations present in viruses means they can develop a mutation that then enables them to cross the species barrier (Walker *et al.*, 2018). All the most recent pandemics mentioned above have come from viruses that have crossed the species barrier. The reason they are so deadly is because they are often asymptomatic and non-lethal in their origin species, consequently they aren't as well studied and more importantly there is no vaccine readily available. Once the virus has been identified as a potential threat and work on a vaccine begins it can take up to 24 months to produce an effective vaccine that offers protection from the virus, but during this time the virus can mutate and produce different strains all requiring different vaccinations. The delay in producing a vaccine can prove catastrophic especially if the virus is airborne, highly virulent and deadly.

As such this research paper will focus on developing a machine learning model that is able to

successfully predict viral mutations and then use that information to determine which viruses are likely to then cross the species barrier into humans and become so-called zoonotic. Accurate predictions will enable science to be proactive in their approach and possibly pre-emptively develop plans for testing and calculating the possible infection rates. Two things which have been shown to be imperative in controlling a viral outbreak. Furthermore, if it is required then they could start early development of a vaccine.

Understanding the mechanism by which viruses infect their host cells and how this translates to species barriers and zoonotic viruses is key to interpreting any predictions made. Furthermore, understanding the viral genome and its accelerated ability to mutate is extremely important when applying the model because it ultimately dictates how soon the virus may gain a mutation that will enable it to cross the species barrier. Darwinian evolution is thought to be the foremost driver of generational viral diversity (Fisher, 1999), this paper postulates there is a discernable pattern in the mutation of the virus and what originally seemed like random mutation, is actually, very controlled and well-orchestrated chaos; all geared towards ensuring the survival of the virus (Fisher, 1999).

The challenge arises in finding and successfully applying this pattern, for example there are between 18,959 and 18,961 nucleotides in Ebola. During one cycle of viral reproduction mutations, viral recombination and post-translational modifications can change the structure and mechanisms of the virus (Lee and Saphire, 2009). Given there are four possible choice of nucleotide from just mutations there are between $1.2x10^{17}$ and $13x10^{17}$ possible viral genetic combinations, this number then increases when viral recombination and post-translational modifications are considered. The astronomical number of possibilities has made it near impossible to find the pattern.

However, with advancements in machine learning (ML) and, in particular, rough set theory and neural networks alongside advancements in computing power it is possible to predict viral mutations with sufficient accuracy. This dissertation suggests that machine learning techniques can be used to determine the underlying pattern, which is ultimately driven by natural selection.

Rough set theory is a method for approximating crisp sets and can is used to discover patterns hidden in data. It can be used for feature selection, feature extraction, decision rule generation and pattern extraction; all attribute which make it ideal for determining an underlying pattern in genetic data (Pawlak, 1998, Bonikowski *et al.*, 1998).

A neural network is a representation of a multi-layered perceptron which uses weightings and activation functions to determine the importance of a feature in a dataset. Based on their type, breadth, depth and level of connectivity they're able to extract patterns from different type of data during training, and once trained they can be used as a prediction tool (Byvatov, E. and Schneider, G., 2003).

Other methods for predicting genetic changes include applying Bayesian methods to single nucleotide variations at gene locus (Morin et al., 2020) and using thermodynamic

measurements of the secondary structures of DNA and/or RNA to predict possible future changes (Hofacker *et al.*, 2002; Barash and Churkin, 2010).

The next chapter will review viral genetics and how they change through mutation. It will then go on to review the current ML techniques that are able to find a pattern in genetic information and then use this information to make predictions. Chapter 3 will then cover the project proposal detailing the planned approach to solving this issue.

Chapter 2

2 Literature Survey

This chapter will explain viral genetics and how they change through mutation. It will then go on to review the current ML techniques that are able to find a pattern in genetic information and then use this information to make predictions. Introducing neural networks and rough set theory regarding the prediction of mutations,

2.2 Viruses

2.2.1 Structure

Viruses are microscopic parasites that cannot survive outside of a host organism and as such aren't strictly considered living organisms (Bamford *et al.*, 2002). They contain some form of nucleic acid (genetic material) either as ribonucleic acid (RNA) or deoxyribonucleic acid (DNA) but never both, this genetic material is then encapsulated by a protein coat (capsid) and then a further lipid coat in some viruses (Bamford *et al.*, 2002). Viruses are classified by three main characteristics; firstly, by the type and size of their nucleic acid; secondly, by the size and shape of their capsid and finally, whether they have a lipid envelope around their capsid (Bamford *et al.*, 2002). (See Fig. 2.1)

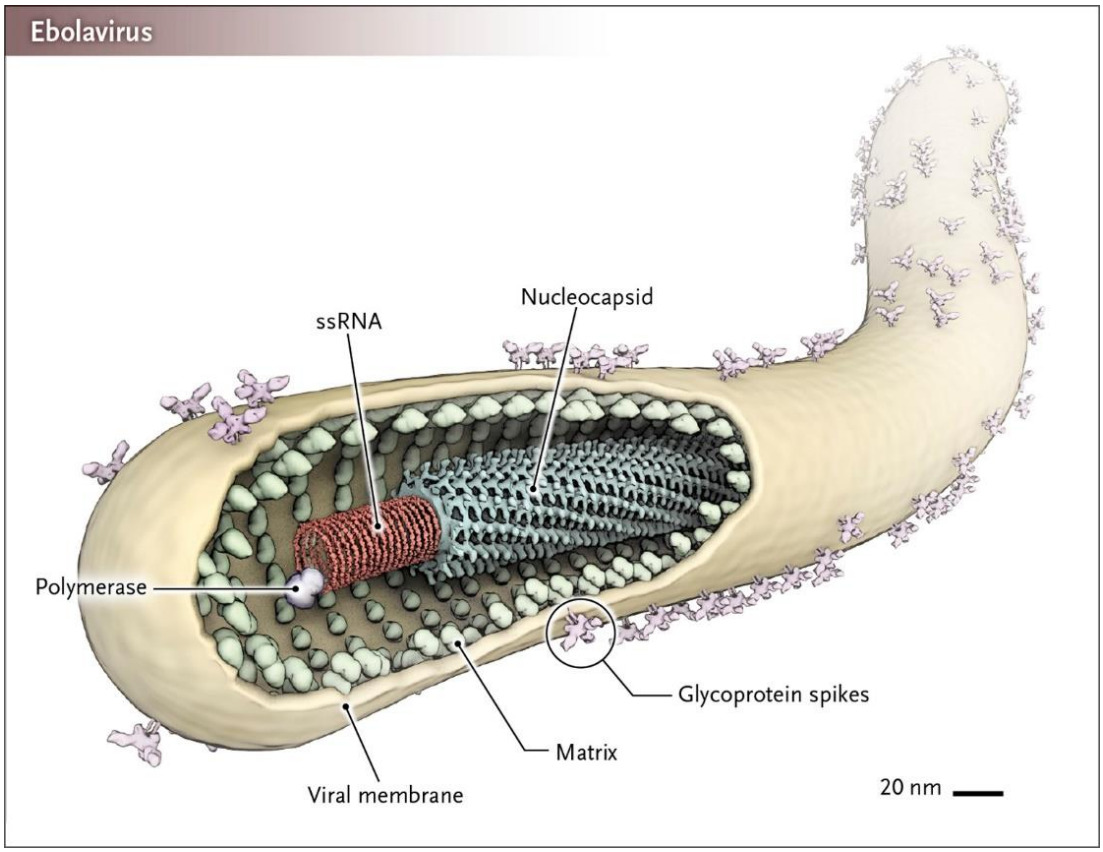


Figure 2.1 A diagrammatic representation of the Ebolavirus. Clearly shows the filamentous shape of the virus, the type of genetic material single-stranded RNA (ssRNA), the nucleocapsid (capsid), some viral proteins (plolymerase) and the outer glycoprotein layer (lipid viral membrane) interspersed with glycoprotein spikes that would be used to infect cells and parasitize them. The figure credit is to (Feldmann, 2014).

The structure of the virus is ultimately decided by the genetic material it has and this structure then informs on the characteristics of the virus. These characteristics include which types of cells it can infect, its virulence, the types of symptoms it produces, how deadly it is and most importantly if it is able to infect more than one species (Isken, 2004).

2.2.1.1 Genetic Material

The genome of any living organism contains within it all the proteins that organism requires to function and survive. This information is coded initially as nucleotides with three nucleotides representing a single amino acid in protein; this is called Triplet hypothesis (Yanofsky, 2007). The viral genome can be either DNA or RNA and then these can further exist as either single-stranded or double-stranded, this refers to how many separate nucleotide strands represent the viral genome (Bamford *et al.*, 2002).

Knowing the type of genome, a virus carries is important because different genomes have different rates of mutation. Single-stranded (ss) RNA and DNA are more likely to mutate than their double-stranded (ds) counterparts, due to variations in their stability. dsDNA is the most stable and least likely to mutate followed by ssDNA, dsRNA and finally, ssRNA which is the least stable and most likely to mutate (Bamford et al., 2002). Furthermore, the type of nucleotides present varies between RNA and DNA (Bamford *et al.*, 2002). With Adenosine , Thymine, Guanine and Cytosine in DNA, and Uracil Adenosine , Guanine and Cytosine in RNA.

2.2.1.2 Mutations and Genetic Variation

A mutation is defined by a random change that occurs in the genetic sequence, either due to an error during replication or because of damage from cellular activity and environment factors such as UV light (Lai, 1992). These changes then introduce variation within the next generation of viruses; however, it should be noted that mutation isn't the only method of introducing genetic variation in viruses. Other methods include genetic recombination, a process during which viruses exchange splices of genetic material after they replication of the genetic material (Lai, 1992) and gene duplications, where there is no change at the original nucleotide site but a gene is duplicated in the next generation resulting in a change of genetic sequence (Gao et al., 2017).

Within genetic mutations there are different types of mutations that can cause changes to the genetic sequence of the virus. (Refer to Fig. 2.2). Point mutations can occur at more than one location during replication of the viral genome resulting in complex changes to the genetic sequence (Hershberg, 2015).

The genetic code determines the structure of protein, as shown in Fig. 2.2 the nucleotides are interpreted in frames of three to give a single amino acid. The order of the amino acids in the protein determines the primary structure, this primary structure then folds into a secondary structure, and finally into a tertiary complex 3D structure with a distinct shape (Orengo *et al.*, 1999). A proteins shape is extremely important because, according to the induced fit, and lock and key hypotheses the shape of protein determines the interactions it can form and for viral proteins this translates to the organism it can infect (Orengo *et al.*, 1999).. Therefore, building a model that can predict changes in genetic sequence can then determine whether a virus is likely to cross the species barrier.

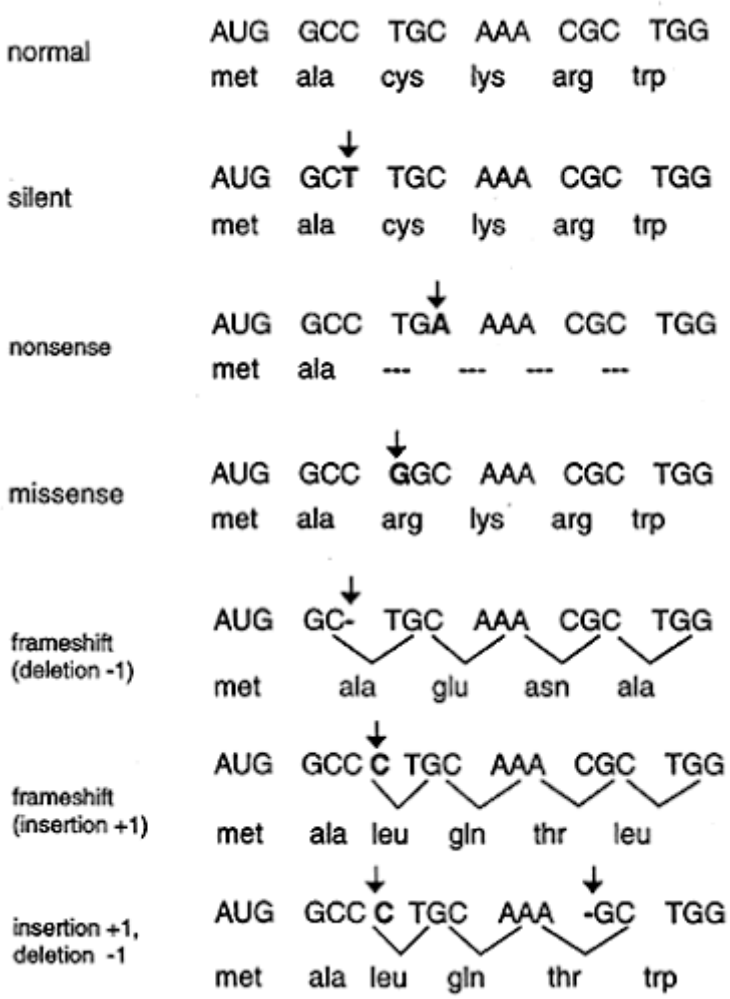


Figure 2.2 A figure showing different types of mutation. The diagram shows various effects of point mutation, on the genetic sequence and subsequently on the amino acid sequence that determine protein structure and function. The original genetic sequence is shown at the top with the various changes and their effects applied to it thereafter. The figure credit is to (Strum, 2020; Hershberg, 2015).

Since viruses are unable to survive outside a host, they need an extremely high rate of mutation because it generates a high level of genetic variety, which makes them extremely adaptable and gives them the highest chance of survival as dictated by the theory of natural selection (Fisher, R., 1999).

Understanding the mechanisms that can affect the genetic sequence of a virus is important because they are the source of the changes in the virus that enable it to cross the species barrier and gives an indication to the difficulty of extracting a pattern and predicting mutation.

2.2.2 Infection Cycle

The infection cycle of a virus is determined by its surface glycoproteins and starts with attachment and entry into the cell; and it is completed when the virus has replicated to a critical point, lysing the cell and releasing more viral particles (See Fig. 2.3) (Yu et al., 2017). If a virus doesn't have the correct glycoproteins it may be able to attach to a cell but not gain access or not be able to attach at all; that's why you have various viral categories (i.e. respiratory, gastrointestinal, immunodeficiency etc.) as they have adapted different glycoproteins and can only infect certain cells inside the body. Logically, these glycoproteins also determine which species a virus can affect and the main determinants of species barriers for viruses (Yu et al., 2017).

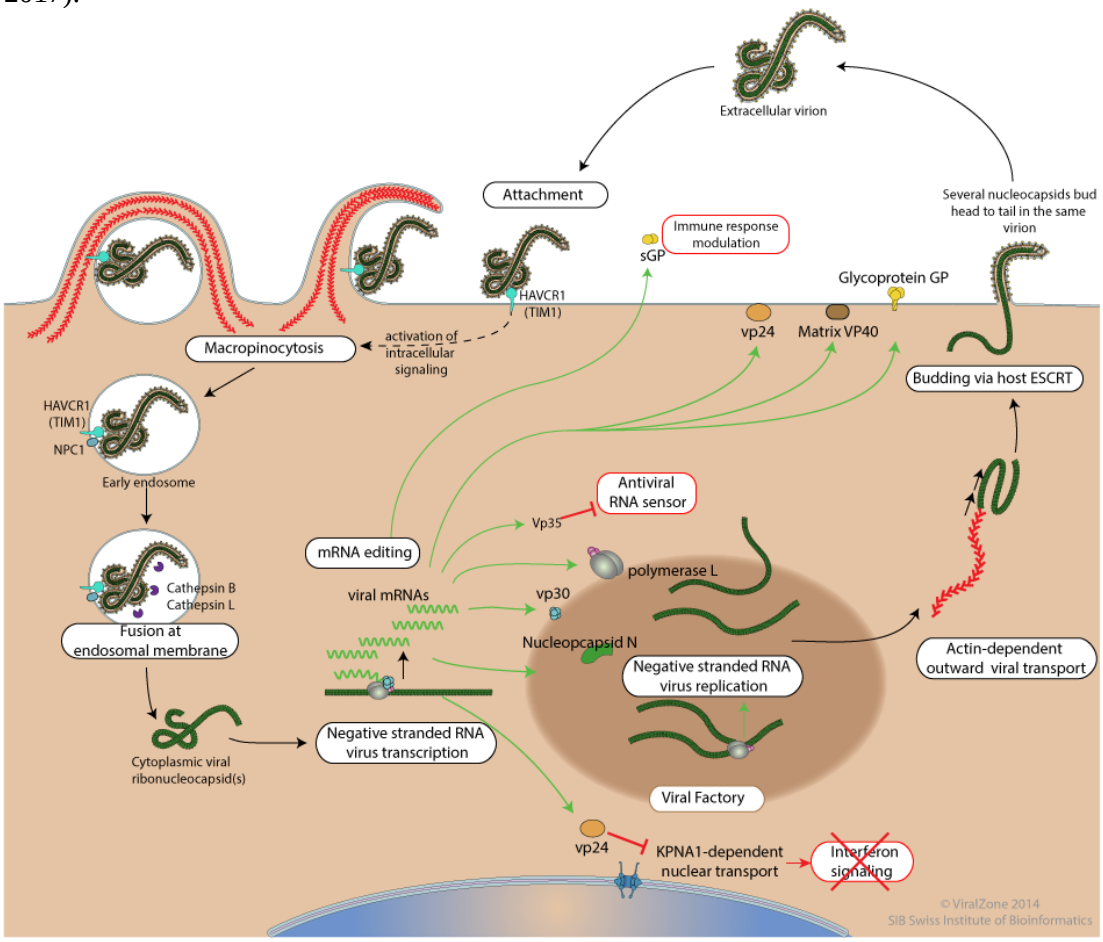


Figure 2.3 A diagrammatic representation of the Ebola infection cycle. Once the virus has attached to, and entered, the cell; it splices its genetic material into the hosts' and then hijacks the hosts' genetic replication mechanism (polymerase) to self-replicate and create more viral particles with which it can propagate its infection of the host. The figure credit is to (Ebolavirus cycle, 2020; Yu et al., 2017).

Artificial neural networks and rough set theory in the prediction of zoonotic mutations in viral haemorrhagic fevers.

The glycoproteins that start the infection cycle and enable a virus to infect a cell are key here and are determined by the genetic sequence. The shape of the glycoprotein will be complimentary to a binding protein on its target cell and if there is any change in glycoprotein shape because of genetic mutation it is complimentary to different binding proteins, possibly enabling a virus to infect a new type of cell possibly in a new species.

2.3 Machine learning and genetics

There are several approaches to using machine learning to predict genetic changes. Natural language processing (Asgari and Mofrad, 2015) in combination with artificial neural networks (Salama *et al.*, 2016; Yin et al., 2020) and rough set theory (Salama *et al.*, 2016) are effective when it comes to predicting changes directly to the genetic sequence. However, other approaches predict the changes more indirectly by applying Bayesian methods to single nucleotide variations (SNVs) at particular gene loci (Goya et al., 2010). This review will describe some of the approaches used in the past, how they work and how they may be applicable to this project.

2.3.1 Natural Language Processing and genetics

Natural Language Processing (NLP) is a field of linguistics that usually studies the interactions between human languages and computers. A paper by Asgari and Mofrad in 2015 showed those same techniques could be applied to interpreting protein and genomic sequences. They showed that representing the genome as a sentence and using NLP to pre-process it into grams of size *n* x *m,* would improve efficiency of training machine learning models (Asgari and Mofrad, 2015).

2.3.2 Artificial Neural Networks

Artificial neural networks (ANN) are computing models based loosely on the biological neural networks observed in the brain (LeCun, Bengio and Hinton, 2015). Given labelled data they can effectively extract a pattern by assigning weights and task-specific rules. They are designed with layers, each layer containing several units in parallel, each unit takes a weighted input, applies a mathematical transformation to it and then outputs it into the next layer (LeCun, Bengio and Hinton, 2015). (See Fig. 2.4)

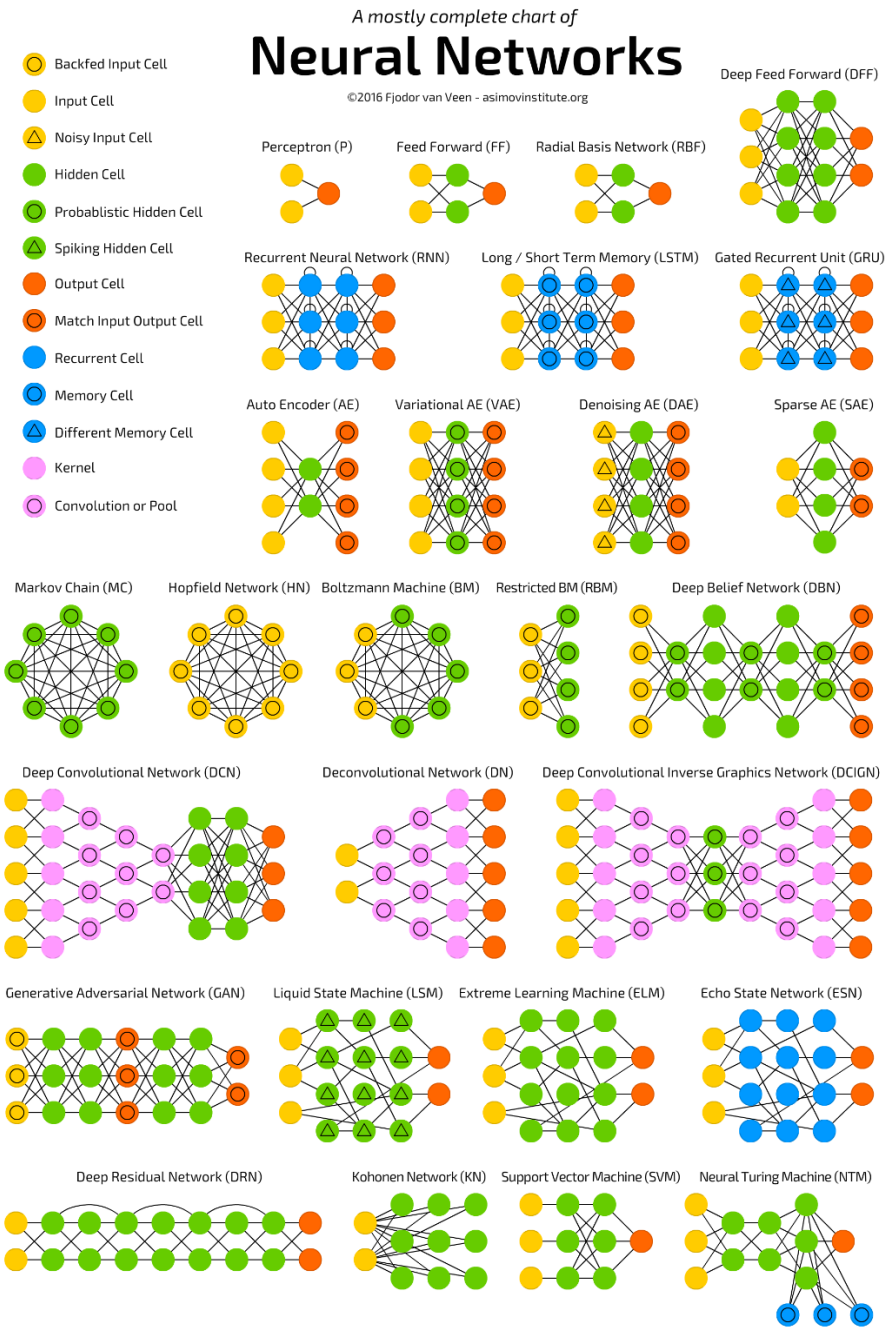


Figure 2.4 Design of different neural networks. Different neural networks work best on different types of data. The architecture of the network (width, depth and shape) is a major determinant on the types of data it can efficiently process. Each network is geared towards solving a specific type of problem presented by a specific type of data input. The figure credit is to (Tch, 2020; Cao *et al.*, 2018).

2.3.2.1 How they work

Artificial neural networks work through supervised learning, labelled data must be input into them during the training phase. During this phase they assign weights to different features of the data, creating classification rules that are never explicitly encoded into the network (Smith, 2017).

2.3.2.1.1 Units

The artificial neural network can be represented as artificial units. In a feedforward network, where every unit in the preceding layer is connected to every unit in the following layer, each connection is assigned a certain weighting during the training phase, this weight is multiplied by the input at the unit and transmitted to the uint in the hidden layer (Smith, 2017). Once a unit in the hidden layer receives all the inputs from its input connections it performs a mathematical transformation through an activation function which determines its corresponding output (Smith, 2017).

2.3.2.1.2 Training

During the training phase labelled examples of the data are input into the neural networks to find the optimum weights for all the connections. The aim of the network is to iterate through all possible combinations of connection weightings until the model function output matches the actual value as closely as possible.

2.3.2.1.3 Activation Functions

An activation function is simply a function that determines the output of a node, they are also called transfer functions (Ramachandran, P *et al*., 2017). They can be split into two main groups Linear and Non-linear activation functions. These functions are applied at each node and dictate what the node transmits (Ramachandran, P *et al*., 2017).

2.3.2.1.3.1 Non-linear functions

Non-linear functions are more widely used, they enable the neural network to finely tune the weight of all the neurons and more importantly deal with the patterns in complex non-linear data (Gulcehre *et al*., 2016). There are four main types of non-linear functions Sigmoid/Logistic, Tanh/Hyperbolic tangent, Rectified Linear unit (ReLU) and leaky ReLU (Gulcehre *et al*., 2016). All the activation functions have their strengths and weaknesses in terms of computation time, ability to generalize, learning rate and whether it is a regression or classification problem (Gulcehre *et al*., 2016). One of the best ways to find which function works best for a neural network is to test them all and track metrics such as computation time, learning rate and accuracy, then select the one which performs best (Ramachandran, P *et al*., 2017).

When considering non-linear functions there are two main characteristics to consider, the derivative and monotonicity. The derivative is important for backward propagation during the training of the weights and the monotonic function simply determines if the activation function

is non-increasing or non-decreasing (Gulcehre *et al*., 2016).

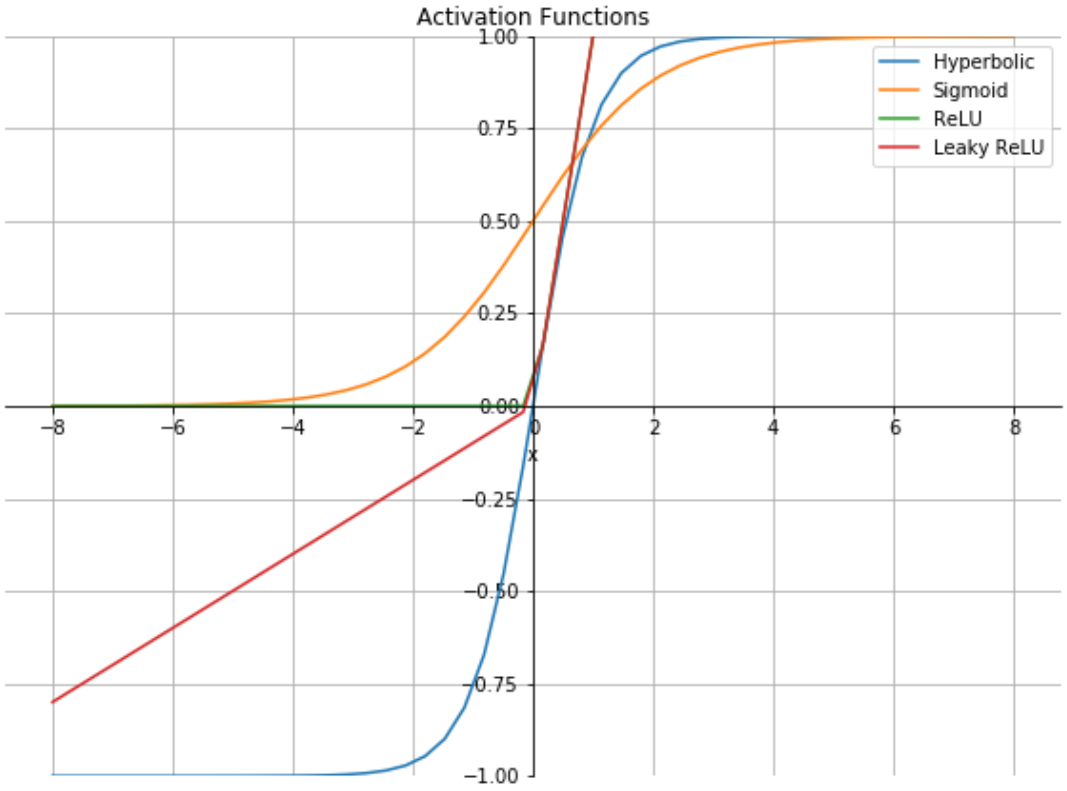


Figure 2.5 Activation Functions. A graph generated in Python 3.7 showing all the possible values for each activation function.

2.3.2.1.3.1.1 Sigmoid function

$$g(z) = 1/1(1 + e^{\{-z\}})$$

As shown in the graph in Fig. 2.5 this functions' values are always between 0 and 1, it is most commonly used on output layers where the expected outcome is best represented as a probability. However, since a large proportion of its gradient is close to zero the it will be harder for the algorithm to determine a pattern and learning rate would be slow (Gulcehre *et al*., 2016).

2.3.2.1.3.1.2 Hyperbolic tangent function

$$g(z) = (e^{z} - e^{-z})/(e^{z} + e^{-z})$$

The hyperbolic function is superior to the sigmoid function because it is centered around zero and its range of values is very small, this translates to faster learning speeds. However, just like the sigmoid function its gradient is near zero over a large proportion of the function which negatively impacts the learning rate (Gulcehre *et al*., 2016).

2.3.2.1.3.1.3 Rectified linear unit function (ReLU)

$$g(z) = \max\{0, z\}$$

The ReLU function is the most widely used activation function and shares a lot of properties with linear activation functions it works well with most problems. As shown in Fig. 2.5 the function has a gradient of zero for $z \leq 0$ and as such can' learn over that range (Gulcehre *et al*., 2016).

2.3.2.1.3.1.4 Leaky Rectified linear unit function (ReLU)

$$g(z) = \max\{a * z, z\}$$

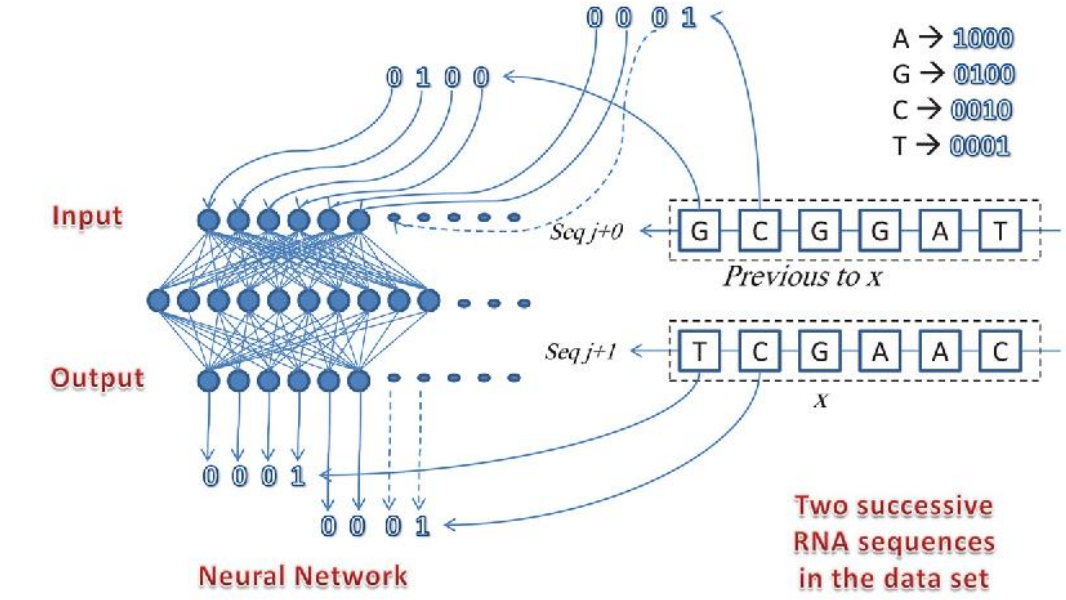
The leaky ReLU is the solution to zero gradient for $z \leq 0$ in the ReLU function. Allows the function to learn for negative inputs (Gulcehre *et al*., 2016).

2.3.2.2 Neural Networks and genetics

2.3.2.2.1 Deep feedforward networks

With advancements in computing power, neural networks are an extremely efficient way to find patterns in non-linear data, making them the perfect tool for finding the pattern in such a complex environment. Work done by Salama *et al.*, in 2016 involved using a deep feedforward (DFF) neural network to predict the viral mutations of the Newcastle virus. Using this the network they were able to obtain a good level of prediction accuracy (Salama *et al.*, 2016).

Since neural networks can only deal with numerical inputs, the nucleotides in the genetic sequence must be numerically encoded. The number of input nodes into their neural network were four times the length of the genetic sequence encoded. This was done because each possible nucleotide had to be one-hot encoded to ensure the distance between nucleotide maintained equal value (i.e. Adenosine [1000], Thymine [0100]) with a set of four input nodes representing each position in the genetic sequence. Using an integer sequence would make the neural network smaller, however using integers to represent each nucleotide (i.e. [0123]) means that the difference between 0 and 1, and 0 and 3 is not equal even though it should be weighted equally. This would mis-train the neural network and negatively affect the model's accuracy. One fully connected hidden layer is used to analyze the sequence (See Fig. 2.6).



neural network from the input data set

Figure 2.6 A neural network for predicting viral mutations. Diagram shows the input layer followed by the hidden processing layer and finally the output layer of next generation. During train the input is the sequence of a one viral generation (j+0) and the labelled output is the next generation (j+1). The figure credit is to (Salama, Hassanien and Mostafa, 2016).

This model is extremely effective because it is extremely versatile and could potentially analyse an entire viral genome. However, the large number of nodes required for even short viral sequences (i.e. 30 base pairs would require 60 input nodes per layer) meaning that it has a high computation complexity and this can make the model both difficult to train in terms of time and optimisation as there are possibly several local minima with an input of that size. To solve this issue Salama *et al.*, 2016 had to make predictions on mini-batch samples of the genetic sequence, and although this produced an accurate prediction (approximately 70%) it can't make connections between all parts of the genetic sequence and thus can't be fully effective.

2.3.2.2.2 Recurrent neural networks

Recurrent neural networks (RNN) are a type of neural network that can store memories of previous events in order to eliminate errors and improve prediction accuracy. A recent study conducted by Rui *et al.*, this year (2020) implements a RNN for the prediction of influenza A mutations using time series data (Yin et al., 2020). The aim of the model is to predict the most likely influenza strain to occur in the following season, the RNN model is implemented with long-short term memory (LSTM) and an attention-mechanism (Yin et al., 2020). (See Fig. 2.7)

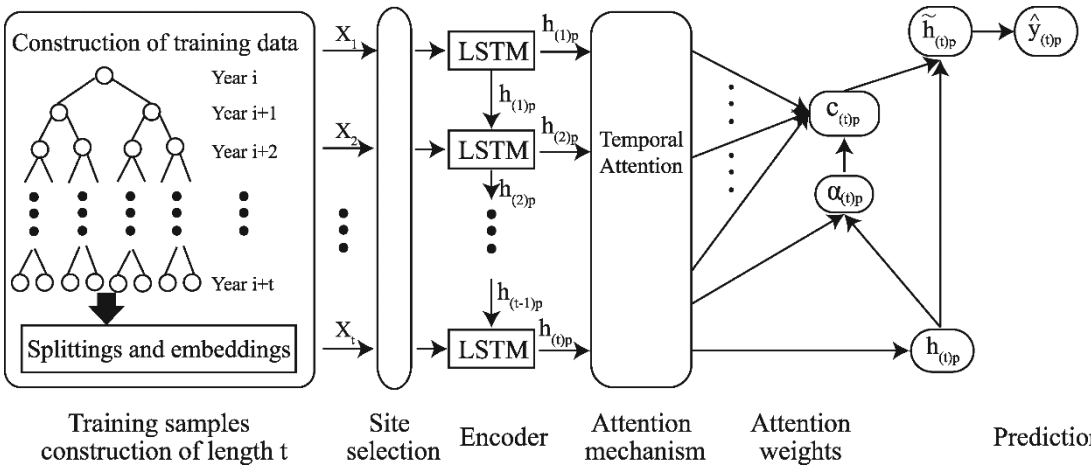


Figure 2.7 A graphical representation of the RNN model used to make the predictions. A) The splitting and embedding of the data are to determine exactly how much variation has occurred since the original strain. B) Site selection determines which site location they want to predict mutations for. This location is then input into the LSTM RNN, its output is processed through an attention mechanism before a prediction is rendered. The figure credit is to (Yin et al., 2020).

The biological sequencing data is first processed using the NLP-based ProtVec method, which provides a representation of the genetic sequence in a form that can be more efficiently processed by the neural network in this case an initial vector of weights (Asgari and Mofrad, 2015). This technique represents the genetic code as several over-layed 3-3-grams, like the kind used in NLP, with the target prediction site in the middle (Yin et al., 2020).

The embedding and splitting of the data contextualise the results; if over time the virus produces more strains producing a wider tree then lower accuracy is not anomalous, because any extractable pattern is more difficult to find. Their model (Tempel) after training was assessed by accuracy, precision, F-measure (sensitivity) and Matthews correlation coefficient (MCC) (Yin et al., 2020). Sensitivity measured the proportion of actual positives that are correctly identified, while the MCC measures the quality of classifications, that is less influenced by imbalanced testing sets since it considers mutual accuracies and error rates on both classes (Yin et al., 2020). Their Tempel model was able to outperform linear regression, plain RNN and support vector machines (Yin et al., 2020).

The use of NLP in the pre-processing of the data accelerates the training process and increases the model's efficiency (Yin et al., 2020; Asgari and Mofrad, 2015). Although their novel approach yields excellent results it may not be applicable to predicting mutations that cause viruses to jump the species barrier. RNN's require large amounts of time series data to be trained effectively and zoonotic viruses only cross between a few species, giving very few time series points. Furthermore, influenza is better studied, documented and sequenced, with sequencing data for individual proteins readily available. Consequently, their predictions can focus on a protein thus greatly reducing the size of the input they have to analyse in comparison to viral haemorrhagic fevers.

2.3.3 Rough Sets

2.3.3.1 How they work

A rough set is defined as a combination of the lower and upper approximations of a crisp set (Han *et al.,* 2012). As a tool for pattern extraction the rough set adds all possible rules that might affect the pattern to the upper approximation, and only adds rules that affect the pattern to the lower approximation of the set (Pawlak, 1998). Applying weights to the rules in the upper and lower approximation sets, they can then be used to create a discernibility matrix, (See Fig. 2.8) that can be used to make predictions based on the rules extracted during training (Pawlak, 1998).

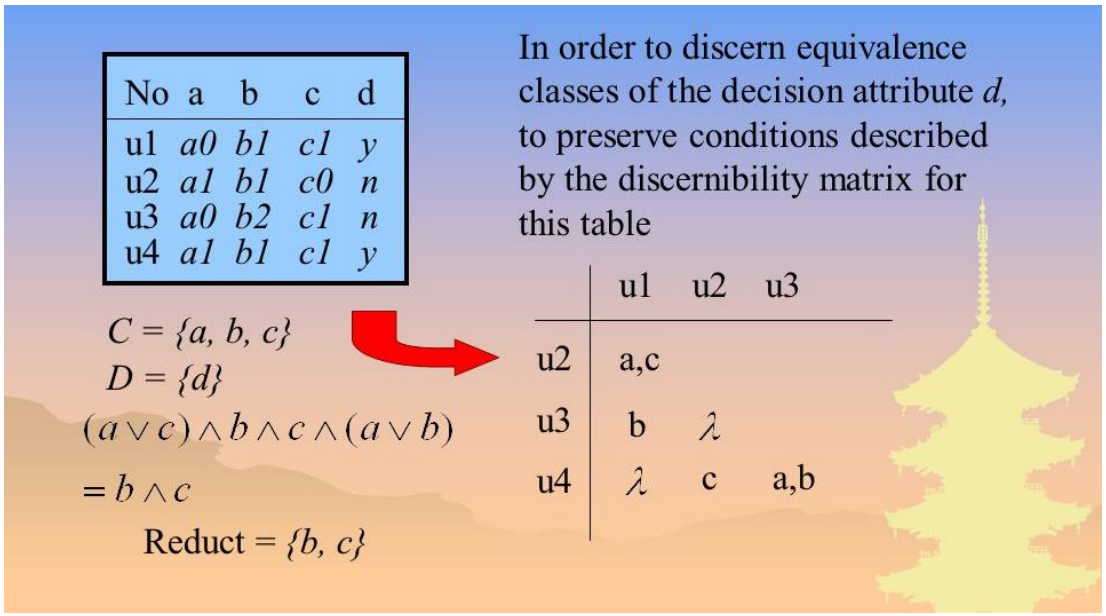


Figure 2.8 An example of a discernibility matrix. Shows a discernibility matrix with the corresponding dataset and how the data would be represented. The figure credit is to (Cauthron, 2020).

2.3.3.2 Rough Set theory and genetics

Working on the same dataset (Newcastle virus) Salama *et al.,* 2016 were able to obtain better results using rough set theory, the rough set gene evolution algorithm they proposed had a prediction accuracy of approximately 75% (Salama, Hassanien and Mostafa, 2016). The model was designed to extract the rules that govern the change at each individual nucleotide in a sequence. (Refer to Fig. 2.9). Furthermore, since there are no numerical parameters to be optimised the rough set genetic evolution algorithm can be given the genetic sequence as letters and coded to account for the four possibilities at each nucleotide position [A,C,T,G].

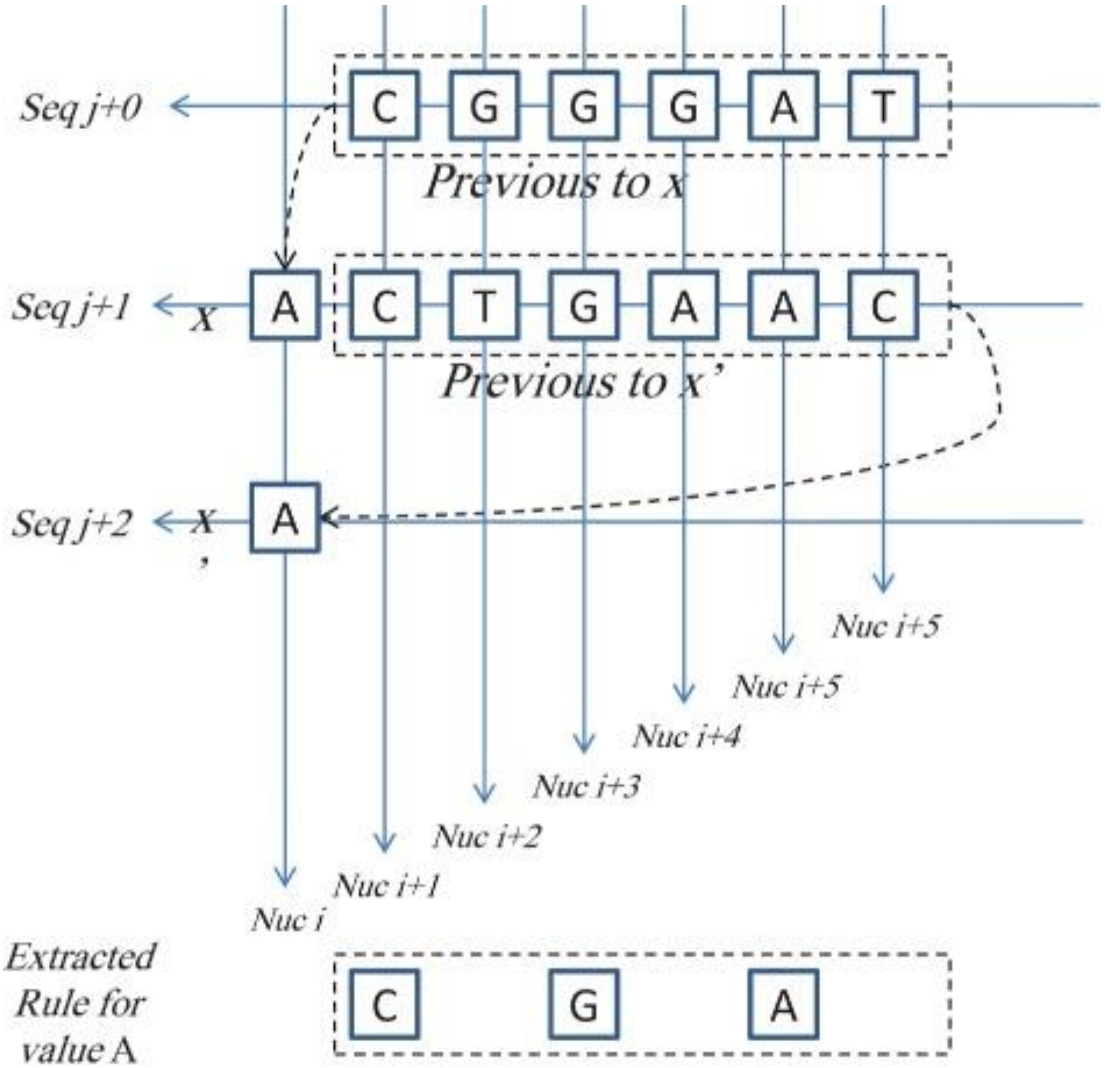


Figure 2.9 Visualisation of the rough set gene evolution algorithm. The tracked nucleotide position (nuc i) is unchanging across generations and as such the algorithm only includes other non-changing nucleotide positions to its rule set. It keeps track of the nucleotide type and its distance from the tracked nucleotide position (i.e. nuc i +2). The figure credit is to (Salama *et al.*, 2016).

The algorithm is input with the genetic time series data with the genetic sequence of the previous generation and next generation given to the model during training. The model then iterates through all the nucleotide positions while simultaneously tracking the rest of the nucleotides in the sequence (Salama *et al.*, 2016). It extracts rules by recording which nucleotides positions mimicked the current tracked position. As seen in Fig. 2.9 tracked nucleotide (Nuc i) doesn't change at either the previous or next generation therefore the model only includes the corresponding nucleotides in the sequence that do not change, effectively linking all the nucleotide positions. Conversely, if the tracked nucleotide were to change across

the generation then the model would include only the changing nucleotides to its rule set and exclude the rest (Salama *et al.*, 2016).

Since the model can take the genetic sequence as letters, the size of the input is greatly reduced compared to the neural network, giving it a computational complexity of $O(N * K^2)$ where K is the number of nucleotides in the sequence and N is the number of sequences. Allowing longer sequences to be analysed and thus patterns to be extracted between more nucleotides (Salama *et al.*, 2016).

2.3.4 Bayesian Inference

2.3.4.1 Bayesian Inference and genetics

Although it isn't a regularly used method regarding genetic predictions, Bayesian inference has the potential to revolutionise the ability to predict mutations, mainly owed to its ability to incorporate beliefs. In theory, the model could be adjusted for as many priors and hyper-parameters as computational power would allow.

The model used by Goya *et al.*, in 2010 to predict the presence of single nucleotide variations (SNV) while reading sequencing data produced by next-generation sequencing (NGS), was able to successfully predict the presence of an SNV in a tumour cell population given the genetic sequence of the surrounding tumours (Goya et al., 2010). An SNV is genomic sequence where a single reference nucleotide changes with unlimited frequency and with varying effects on the genotype depending on the change (Goya et al., 2010). The genotype of a gene in DNA determines whether both strands of DNA carry the same copy of the gene and is an important indicator in predicting genetic changes. They used the expectation maximisation (EM) algorithm to find the maximum a posteriori (MAP) to estimate the parameters given a training dataset (Goya et al., 2010).

The model takes a single nucleotide and aligns it with sequencing data from several different cells, it then compares each nucleotide in the sequence in reference to a SNV and if it has exhibited any changes between sequences of different cells (Goya et al., 2010). This is the data that is used to train the model, so in theory once training is completed and the model is given a standalone NGS generated sequence it can generate a distribution for the likely changes at that SNV and thus sequences likely to be found in surrounding tumours (Goya et al., 2010). The model was evaluated on DNA extracted from ovarian tumours with the conditional probability distributions for the parameters set as follows. (Refer to Fig. 2.10)

$$
\begin{aligned}
p(\pi|\delta) &= \mathrm{Dir}(\pi|\delta) \\
p(G_i|\pi) &= \mathrm{Multinomial}(G_i|\pi,1) \\
p(a_j^i|G_i{=}k,\mu_k) &= \mathrm{Bern}(a_j^i|\mu_k) \\
p(a^i|G_i{=}k,\mu_k,N_i) &= \mathrm{Binom}(a^i|\mu_k,N_i) \\
p(\mu_k|\alpha_k,\beta_k) &= \mathrm{Gam}(\mu_k|\alpha_k,\beta_k) \\
p(z_j^i) &= \mathrm{Bern}(z_j^i|0.5) \\
p(q_j^i|a_j^i,z_j^i) &= \begin{cases} q_j^i & \text{if } a_j^i{=}1, z_j^i{=}1 \\ 1-q_j^i & \text{if } a_j^i{=}0, z_j^i{=}1 \\ 0.5 & \text{if } z_j^i{=}0 \end{cases} \\
p(r_j^i|z_j^i) &= \begin{cases} r_j^i & \text{if } z_j^i{=}1 \\ 1-r_j^i & \text{if } z_j^i{=}0 \end{cases}
\end{aligned}
$$

Figure 2.10 Conditional probability distributions for the SNVMix model. Figure credit to (Goya et al., 2010).

Parameter	Description	Value
δ	Dirichlet prior on π	(1000,100,100)
π	Multinomial distribution over genotypes	Estimated by EM (M-step)
G_i	Genotype at position *i*	Estimated by EM (E-step)
a_j^j	Indicates whether read *j* at position *i* matches the reference	Observed in SNVMix1, latent in SNVMix2
z_j^j	Indicates whether read *j* aligns to its stated position	Latent
q_j^i	Probability that base call is correct	Observed (SNVMix2 only)
r_j^i	Probability that alignment is correct	Observed (SNVMix2 only)
μ_k	Parameter of the Binomial for genotype *k*	Estimated by EM (M−step)
α	Shape parameter of Beta prior on μ	(1000,500,1)
β	Scale parameter of Beta prior on μ	(1,500,1000)

Table 2.1 Definitions of values used in the equations in Fig. 2.10 above. Figure credit to (Goya et al., 2010).

Each parameter is given its own conditional probability distribution with a mixture of beliefs, some that are to be observed from the NGS data such as the read matching the reference, the probability the base call is correct and the probability the alignment used for the comparison is correct. Those distributions that aren't observable and analytically intractable are estimated using the EM algorithm (Goya et al., 2010).

Once the model is trained and the parameters optimised it was able to improve the accuracy on the accuracy of a prediction by approximately 4% on any given sequence compared to the previous models (Goya et al., 2010).

Although this model was able to improve on the accuracy of previous models it has two main issues. Firstly, because it can only focus on a single nucleotide location it means you must focus on specific protein gene sequences only and this would involve identifying the proteins that enable viruses to cross the species barrier and their key genetic loci. This presents new challenges and isn't efficient especially when those proteins maybe different across different viral species. Secondly, because it incorporates the genotype of the given gene into the priors it is unclear how it will function on RNA viruses because genotypes are more informative in DNA (current model) than in RNA (proposed viral model) (Darrier et al., 2019). Given that genotyping can provide a large amount of information as a prior this model would be less accurate when applied to RNA viruses.

Chapter 3

3 Data and Methods

3.2 Data

Data used in the training, validation and testing of the model was sourced from Uganda Virus Research Institute (UVRI) being a mixture of data collected personally and data they had on hand. The data is the coding regions of several proteins in the various haemorrhagic fevers. The sequences vary in the type of sample fluid the virus was collected from, the species of animal they were collected from and the time the generation (year) they were collected. Genetic sequencing was done using the Sanger method (Sanger, Nicklen and Coulson, 1977).

3.2.1 Varying virus

Due to the lack of data on any given single virus, a mix of viral haemorrhagic fevers were used to give a greater number of training example and improve the accuracy of any model then created. The similarities in infection cycle, type of cells infected, and symptoms observed in infected individuals is the viruses are grouped as haemorrhagic fevers. These characteristics mean they affect the infected organisms in similar ways and as such will contain similar proteins.

3.2.2 Sample Fluid

The variation in sample fluid dictates which fluid the proteins was isolated and sequenced from options being saliva, mucus, blood plasma or CNS. The sample fluid the protein was isolated from does not usually affect the viral proteins unless they are environment specific binding proteins. i.e. proteins which have evolved to bind to specific cells in specific areas of the body.

3.2.3 Varying Host Species

The variation in viral genes between species would be the greatest. As viruses infect different species, they adapt to their host so they can most efficiently evade their immune response; penetrate, survive and reproduce within the hosts' cells and propagate their infection. These adaptations come in the form of changes to viral proteins which in-turn originate from changes in the genetic sequence and as such varying host species show varying genetic sequences for the same viral proteins.

3.2.4 Generations

Viral generations in this case were defined by the year the viral sample was collected. Any variation between viral samples collected in the same year was considered to be inconsequential and therefore negligible.

3.2.4.1 How it works

Bayesian inference is a method of statistical inference that employs Bayes theory and allows for prior beliefs or observations to be incorporated into the predictions. It also allows for the

update of this predictions given new beliefs or observations (Gelman et al., 2015).

3.3 Preprocessing and Initial Analysis

Data was sent as a fasta file containing all the protein sequences separated by viral species. The fasta file format is used in bioinformatics and biochemistry to represent nucleotide or amino acid sequences using letters. The format also allows for sequence names, date of collection and comments to precede the sequence. Data was extracted and contained as a list of a list. An initial analysis of the genetic data showing the different lengths and the average similarities between sequences is shown in Fig 4.1 below.

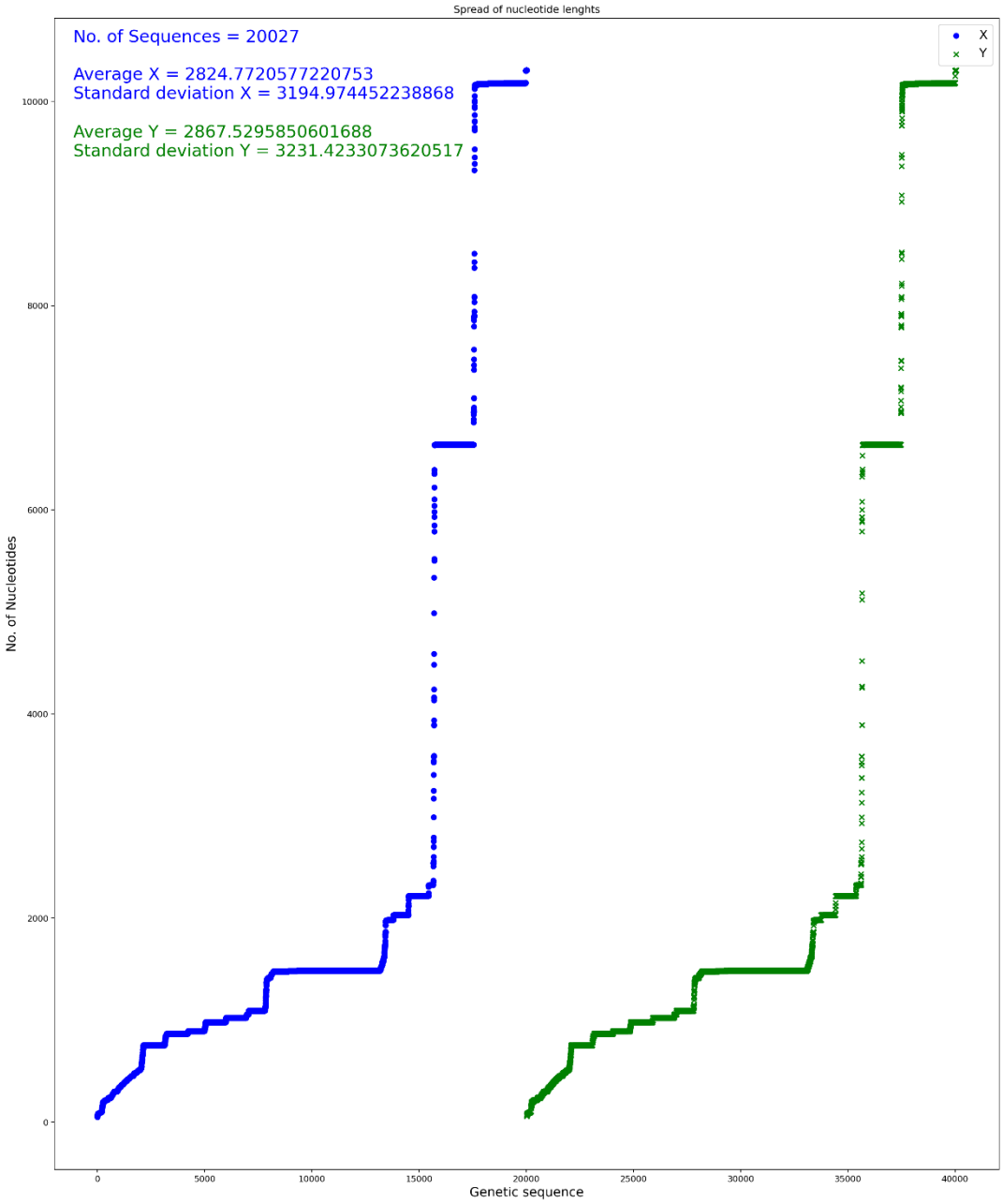


Figure 3.3.1 Compilation of viral data in the input and target datasets. The metrics show the average length of a genetic sequence along with the variance of viral genetic sequences.

Each individual viral species genome was independently imported and split into training, validation and test set to ensure that the input and target data examples were of the same species and as such minimized the effect of varying nucleotide length on the training of the model. The resultant data splits were then further split into input (X) and target datasets (Y), with an average difference in length between input and target sequences of 14 base pairs. The sequences were also aligned so they always started with the initiation codon 'ATG' to ensure that they were all in the same reading frame, keeping them in the same reading frame ensures the nucleotide sequences are consistently read from the first amino acid they code to the last.

Even though the data was only coding regions of the proteins, it included non-functional coding regions of the protein which are required for the stabilization of proteins during assembly, protein sequences such caps and poly-A tails. In each nucleotide sequence these have the possibility of being different lengths and as such the data had to be put into a standardized reading frame before it could be used for training. Before the assembly of the functional protein section begins the triplet codon 'ATG' is used to signify a Methionine and the start of the actual protein. The end of a genetic sequence is signaled by a stop codon, this can take three forms: 'TAG', 'TGA' or 'TAA' due to redundancy in the genome, the rest of the codons encode amino acids (See Fig. 3.2).

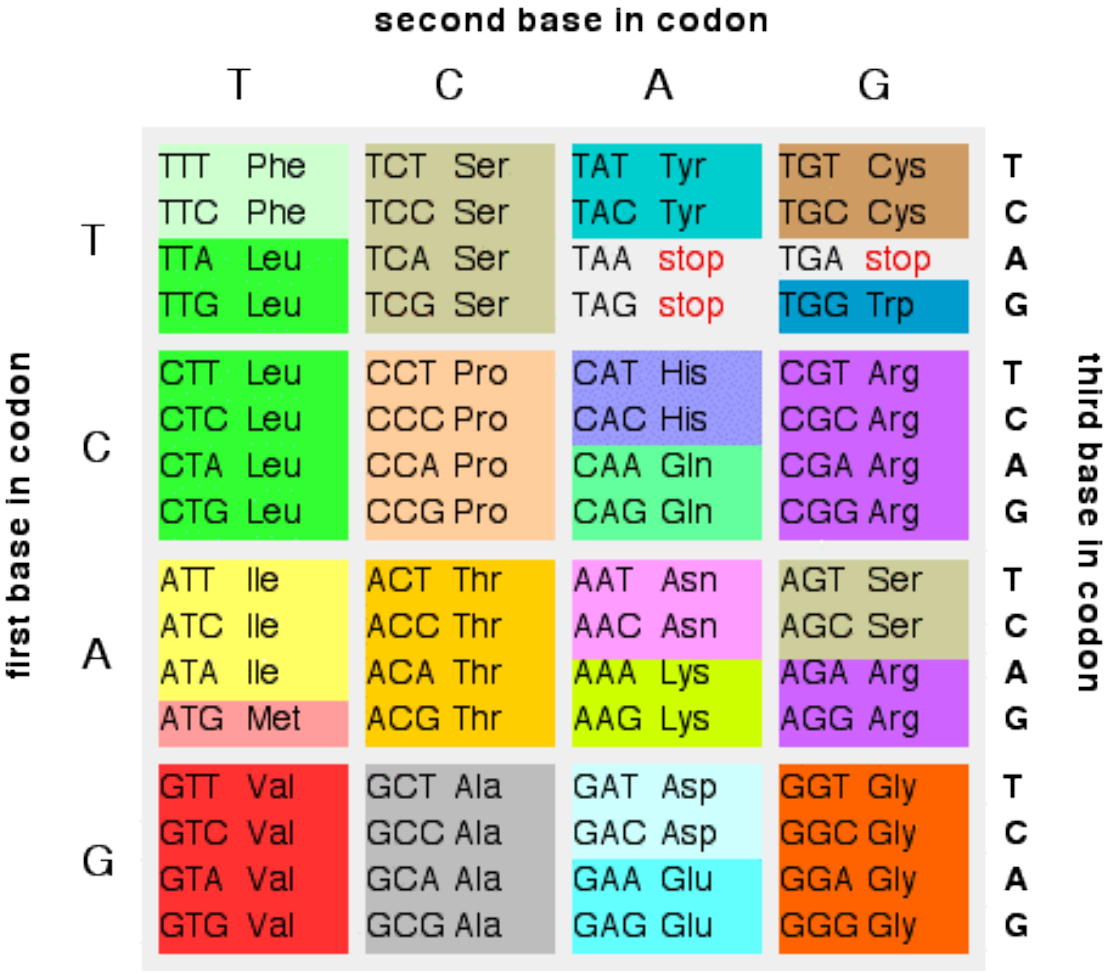


Figure 3.2 DNA triplet codons and their translations. 'ATG' (Methionine) represents the start of all protein sequences. With 'TAA', 'TAG' and 'TGA' signalling the end of protein sequences.

Therefore, to reduce on the variation in nucleotide sequence length, only the sequences between the start and stop codons were considered. Thereafter, the data required more filtering to ensure that only 'A', 'C', 'G' and 'T', the encoding nucleotides shown in Fig, 3.2, were contained in the nucleotide sequences for one-hot encoding and training.

Before the data was ready for one-hot encoding it had to be converted from one long string such that each nucleotide was considered a standalone string for one-hot encoding with 4 possibilities to be considered at each nucleotide position. Or alternatively triplet codon hypothesis could be used, and every three nucleotides could be considered for one-hot encoding with 64 possibilities at each triplet codon position.

Successfully extracting the features that drive viral mutation and applying them to make accurate predictions would be difficult due to the extremely close similarities (shown in Fig 3.3 below) between the X and Y datasets. Therefore, the risk of over-fitting was high. A similarity ratio which calculates shows the proportion of nucleotides that are the same in a pair of sequences is calculated. The average similarity ratio between X and Y sequences in this dataset is 0.61.

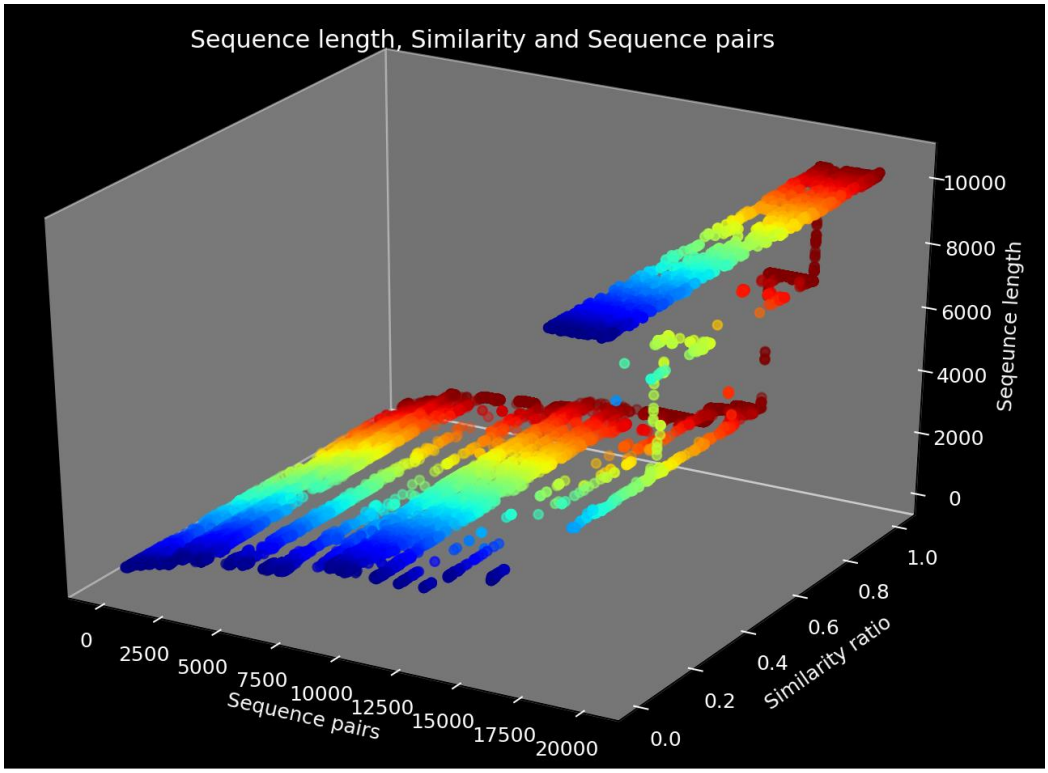


Figure 3.3 3D graph illustrating the relationship between sequence length and similarity between X (input) and Y (target) data.

3.4 Methods

In the previous section the nucleotide Three separate methods are used to make predictions on the nucleotide sequences. Feedforward neural network, long-short term memory recurrent neural networks and a rough set theory algorithm. This section outlines how they are implemented.

3.4.1 Feedforward Neural Networks

As a baseline approach feedforward neural networks were trained to output predicted nucleotide sequences. This was the approach proposed in Salama *et al.,* 2016. The network needs to have as many input units as there were nucleotides in the in the sequence and equally as many output units. This is implemented from scratch in this project using the TensorFlow (Keras) library.

3.4.1.1 Data Pre-processing

After the data was split into, training, test and validations sets it was then one-hot encoded, placed in a 3D-array dictated by [number of exemplars, length of one hot encoding, number of type of nucleotide sequence] and in this form, it was ready for the deep neural network. Given the feedforward network had to have the same number of units as nucleotides in the sequence the number of virus types was always ‘one’ to ensure there was the same length sequences across the entire dataset.

3.4.1.2 Model

With the feedforward neural network, the size of the input had to match the number of units and as shown in Fig 3.1 above there were variations in nucleotide length across the entire dataset, therefore every time the nucleotide sequence length changed the model had to be adapted and changed.

Following on Salama *et al.* 2016, one hidden layer was included with at least the same number of units as the input and output layer. With the longest sequences at just below 3500 nucleotides as seen in Fig. 3.1, the largest hidden layer used is 4096 units. Stochastic gradient descent is used as the optimizer in combination with a mean squared error loss function, the model is run for 4 training iterations (epochs).

3.4.2 Recurrent Neural Network

To improve the initial approach taken by Salama et al., (2016) a recurrent neural network (RNN) could be implemented. More specifically a long-short term memory (LSTM) RNN could be used (Klein et al., 2017). With the advances in natural language processing (NLP) if one considers different generation of a viral genome to be sentences in different languages and the nucleotides to be the words, then the power of NLP can be applied to genetics. This means that the genetic code can be processed as letters directly and variations in sequence length are automatically incorporated into the model.

LSTM models with the ability to store, recall and learn from previous memories are extremely

powerful tools in predicting genetic mutations. With their ability to link different nucleotide positions through their through their memory, the model should be able to more accurately predict viral mutations with more flexibility towards variations in the sequences be it viral species, the host the sample was collected from, the sample fluid the virus was isolated from or variations in sequence length. Furthermore, given the similarities between input and target data shown in Fig. 3.2 the vanishing gradient problem would be an issue and LSTM's solve this issue.

To implement the LSTM a PyTorch library called OpenNMT is used (Klein et al., 2017). Open Neural Machine Translation (NMT) is a library that enables users to build an array of translation models. The model that takes a text file as an input. Each line of the text file is considered an exemplar (sentence) and the end of the sequence (EOS) is denoted by a '\n' newline marker, while spaces in the sentence denote different words.

Once the data had been pre-processed, it is used to train the model and output a fully trained and optimized LSTM model with the ability to take the coding nucleotide sequence of a haemorrhagic fever and predict the mutations that would take place on it in the next generation. This model can then be used as an input into the translation function which renders the predicted nucleotide sequence given an input.

3.4.2.1 Data Pre-processing

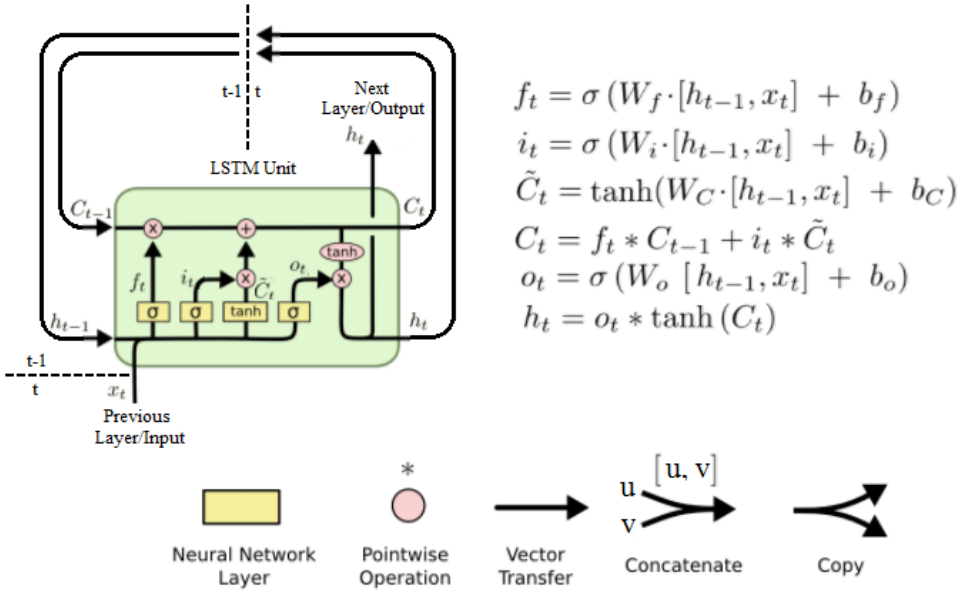
Although the OpenNMT provides a data pre-processing function before that function can be used the nucleotide sequences must be exported as text files. If the nucleotide sequences are to be considered as sentences made up of tokens, then they can be read in three different forms. First, the entire nucleotide sequence can be made as one solid string and considered a word on its own. Second, a space can be included between each nucleotide or, third, one can utilize triplet codon hypothesis and have spacing after every three nucleotides.

The preprocessing tool provided by the library converts the input text filed into a readable 'vocab' file for training. During this pre-processing, the text file is searched for all possible tokens and the number of exemplars that are present. The settings allow for a max sequence length to be set and a max vocab size to be set, this ensures that overly long nucleotide sequences do not affect the model training negatively. The preprocessor also adds embeddings and padding to the text converting it into a numerical vector that can then be an input into the LSTM model.

3.4.2.2 Model

OpenNMT provides several parameters that allow for the fine tuning of the final model. A good starting point was with the default parameters.

Using the equations detailed below in Fig. 3.4, during the forward pass each LSTM unit in the hidden state uses a combination of tanh and sigmoidal activation functions to decide whether that specific vector point should be stored for use in subsequent layers or should be discarded.



$$f_t = \sigma\left(W_f \cdot [h_{t-1}, x_t] \; + \; b_f\right)$$
$$i_t = \sigma\left(W_i \cdot [h_{t-1}, x_t] \; + \; b_i\right)$$
$$\tilde{C}_t = \tanh(W_C \cdot [h_{t-1}, x_t] \; + \; b_C)$$
$$C_t = f_t * C_{t-1} + i_t * \tilde{C}_t$$
$$o_t = \sigma\left(W_o \; [h_{t-1}, x_t] \; + \; b_o\right)$$
$$h_t = o_t * \tanh\left(C_t\right)$$

Figure 3.4 Diagram of an LSTM cell, with representative equations. C_t represents the cellular input, this is the memorised vector input. h_t represents the input from the previous hidden layer. X_t is the new input from the data set. f_t is the vector that initially updates the cells memory after combining the new input data and data from the previous hidden layer using a sigmoid function. i_t and $\check{c}_t$ are multiplied and update $\tilde{C}_t$ using a combination of the sigmoid and tanh activation functions before it is passed to the next layer or output. Finally, the hidden state to be passed to the next layer is updated. (Figure modified from Understanding LSTM Networks -- Colah's blog, 2020)

Given that LSTM's undergo multiple calculation per unit, they have a higher computation complexity and time than vanilla RNN's. They also require more storage space to store their intermediate states and calculations during each forward and backward pass. The OpenNMT library processes sentences as tokens (words) and as such long exemplars are input as large vectors and will deplete the memory if there are too many in a single batch. Therefore, the batch size and validation batch size must be tightly controlled.

For measurement of accuracy a log file is outputted to show the training and validation accuracy at each training iteration. This file is then processed, and the numbers extracted to give the changes in accuracy over training iterations.

This novel approach, in concept and application, could prove extremely useful in providing predictions for viruses with few nucleotide sequences available.

3.4.3 Rough Set Theory

Rough Set Genetic Mutation algorithm used is the one detailed by Salama et al., (2016) and the details can be found there (Salama et al., 2016).

3.4.4 Recall Metrics

The main metric to be measured from the data is the accuracy and this can be measured by doing a nucleotide sequence comparison and producing the percentage of nucleotides that are similar. The OpenNMT model provided this function inbuilt for both training and validation and could be output as a log text file. It also offered other metrics such as the prediction perplexity, which is computed by the equation below with the loss being the cumulated negative log likelihood. This metric dictates the precision of the model and is a confidence score on the model's ability to generate those specific target words.

$$e^{-\frac{1}{N}\sum_{i=1}^{N} ln\, q(x_i)}$$

During translation using the OpenNMT, a beam search algorithm is used and recalls accuracy, a predicted score, predicted average score and the prediction perplexity. The beam search is a heuristic search algorithm that expands the most promising node and in doing so reduces the memory requirements of model. The predicted score is the cumulated log likelihood, the predicted average score is the log likelihood per generated words and the predicted perplexity is defined as exp $(-predicted\ average\ score)$.

During parameter optimization for the LSTM, training accuracy and perplexity in combination with the validation accuracy and perplexity are used as a guide in optimizing the parameters of the LSTM model. The aim is to maximize accuracy with a maximum possible score of 100% and a minimum score of zero, while simultaneously aiming to get perplexity to a score of positive one.

Chapter 4

4 Experiments and Results

This section describes the experiments used to optimize the prediction of nucleotide sequences an OpenNMT based LSTM model. It then describes the results obtained from the optimized model and compares them a feedforward neural network and rough set genetic mutation algorithm (Salama et al., 2016).

4.2 Nucleotide Sequence Format

Determining the format of the nucleotide sequence is important because each format type has unique challenges. Having the nucleotide sequences represented with each individual nucleotide position separate greatly increases the number of tokens per exemplar, however there are only 4 possible variations at any given position, and this greatly reduces the perplexity the model faces. With regards to the feedforward neural network it would increase the number of input and output units required and thus the computational complexity. Similarly, with rough sets the computational time would increase. With the LSTM model this produces exemplars that are long, up to 3,500 tokens long, this means that for every layer of processing, storing the input vector values requires large amounts of dedicated graphics processing unit (GPU) memory and as such limits the width and depth of the model as well as the batch sizes for training and validation.

To solve this issue the nucleotide sequences can be processed as triplet codons and given this is the format that is used to translate the genetic sequences into proteins, any change in the triplets is more likely to translate to protein level changes and non-silent mutation. This reduces the size of the input by one third and consequently the computation time and storage space required. However, the number of possible codon combinations rises from the four possible nucleotides to 64 possibilities and the length of nucleotide inputs can still be up to 1,165 tokens long. This length of input is still restrictive and the maximum size of the neural network, training batch size and validation batch size are still limited. This format of nucleotide sequence, although able to produce high accuracy during training, caused the LSTM model to over-fit and perform poorly during validation and testing. Hence, without a larger training dataset this format of nucleotide sequence is unviable. (See Fig. 4.1 below)

The number of nucleotide sequences available is a limiting factor; with only 20027 nucleotide sequences to train, validate and test the model, low test and validation accuracy due to insufficient training data is a major factor. Data shuffling when creating the training, validation and test splits, along with dropout are used to correct for over-fitting, but the size of the dataset is the main factor in contributing to over-fitting in this scenario.

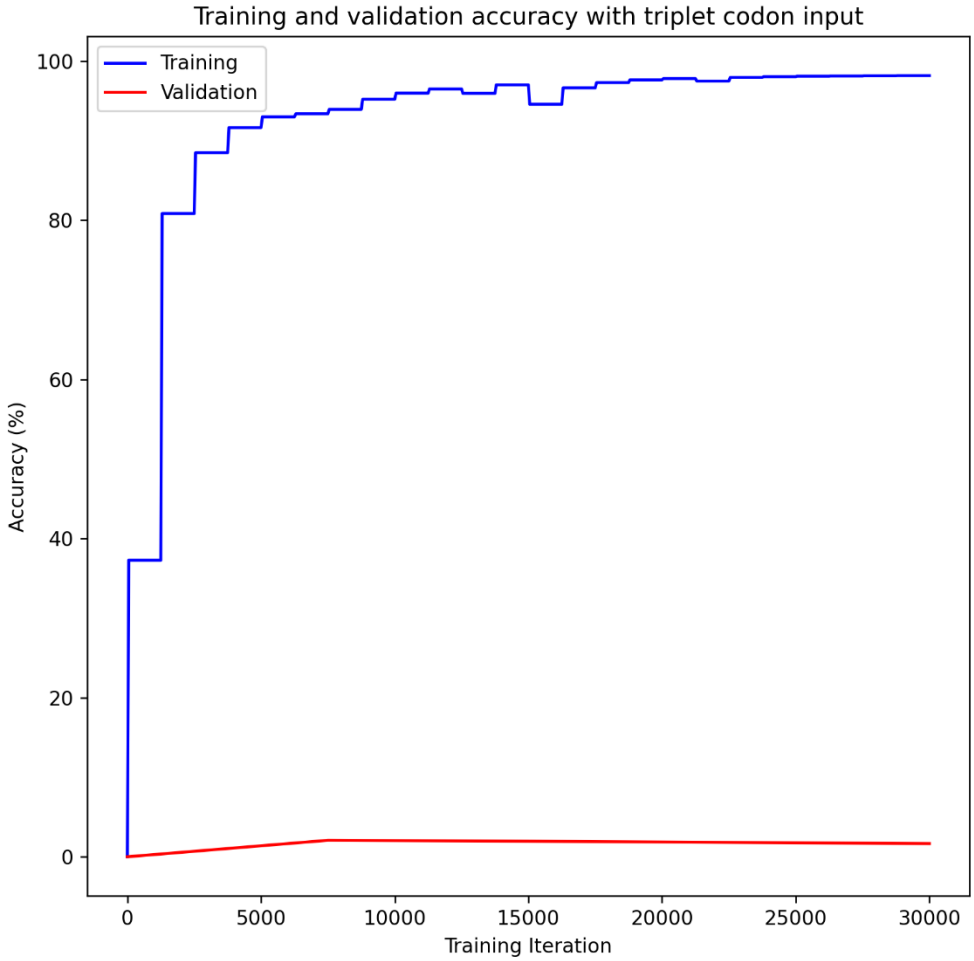


Figure 4.1 Difference between validation and training accuracy when triplet codon format is used. A smoothing function is used on the training accuracy with averages taken every 25 iterations. Graph clearly demonstrates an overfitted model. (8 Layers, 600 Units, Dropout 0.6, learning rate 1).

Further experimentation found these issues could be solved by using a non-spaced version of the nucleotide sequence, with the entire nucleotide sequence considered to be a single token, of varying length. In terms of translation this would be akin to a word for word translation as opposed to sentence-based translations. This causes a reduction in training accuracy however, it improved the validation and prediction accuracy of the LSTM model.

4.3 OpenNMT

Through research the optimum settings for training the LSTM model were found. The number of layers, number of units, learning rate, dropout and method of optimization were all initially set through online research. OpenNMT has been previously implemented as a tool for language translation and the parameter settings and outcome results in those scenarios was a good guide for initial settings (Wu et al., 2016; Gulchere et al., 2015, Klein et al., 2017; Van Noord, R. and Bos, J., 2017, Gehrmann et al., 2018). Granted, this project is working with nucleotide sequences and language translations, some further experimentation is required to achieve maximal training and validation accuracy along with a training and validation perplexity as close to one as possible.

The main parameters that require optimization are learning rate, method of optimization, dropout, batch size (validation and training) and LSTM units and layers with the rest of the parameters being relatively unchanged across other researched models. A grid search is implemented to optimize these parameters.

4.4 Results

The LSTM model can obtain a training accuracy of up to 52.24%, a validation accuracy of 51.7% and validation perplexity of 57.70. over 30000 training iterations. (Fig. 4.2A/4.2B below). The validation accuracy (4.2B) and perplexity (4.2C) were only reported for the model used for translation (10 Layers/512 units/Dropout 0.01/LR 0.1) because perplexity can range from 1 to infinity and thus it is not viable to graph all the outcomes of the experimental models.

The 30 layers/1000 units/Dropout 0.2/Learning rate 0.005 model was the largest model that could run with the computing power that was available and was the initial starting point. The aim from then was to reduce the runtime of the model and increase its efficiency. Several models were tested but the ones in Fig. 4A demonstrate a gradual reduction in model size, increase in learning rate and decrease in dropout to produce a smaller model with 10 layers/512 units/Dropout 0.01/Learning rate 0.1. This model achieves higher training accuracy in fewer training iterations and produces a better overall training accuracy.

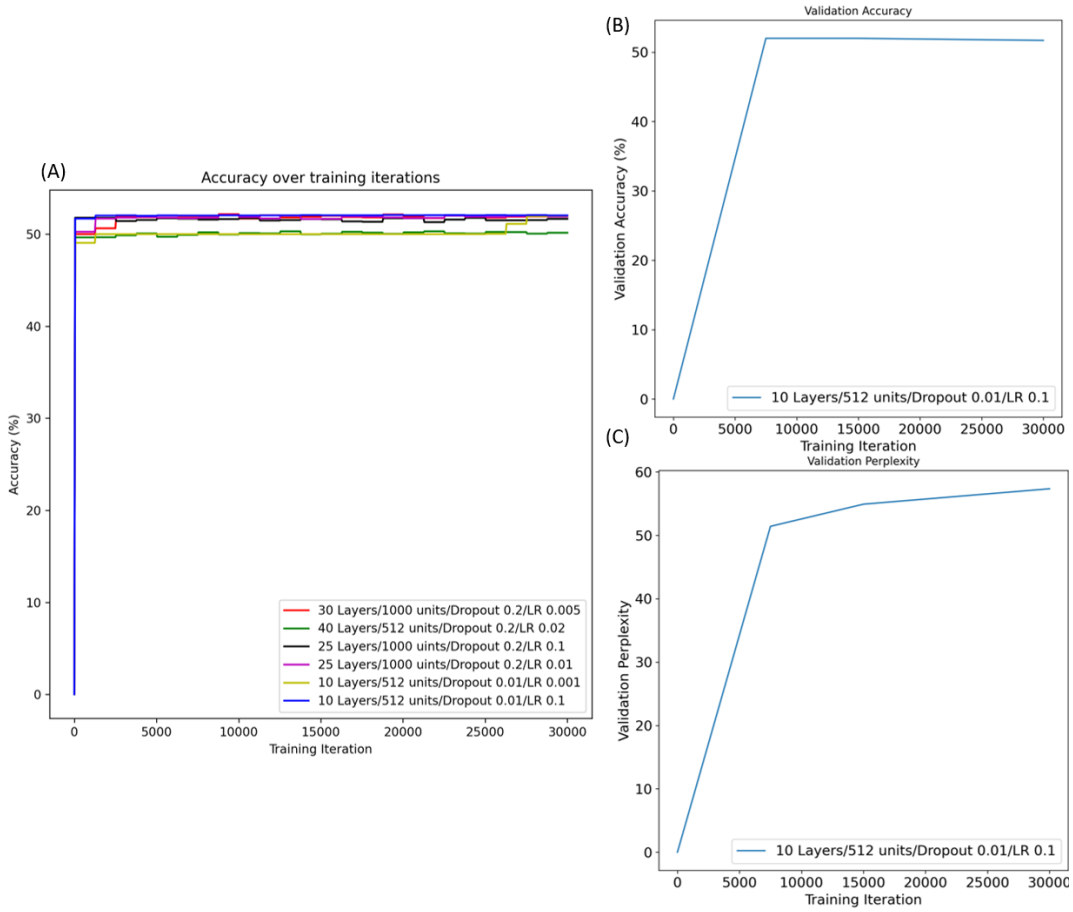


Figure 4.2 Training and Validation accuracy and validation perplexity for the LSTM model. (A) Shows the outcome of the training accuracy utilising different model settings with a smoothing function applied averaging results every 25 iterations. LR – Learning rate (B) Shows the validation accuracy, validation is performed every 10,000 iterations. (C) Shows the validation perplexity with perplexity being a measure of model precision and confidence. Validation is performed every 10,000 iterations.

In comparison the feedforward neural network only required five epochs to outperform the training and validation accuracy of the LSTM model (See Fig. 4.3) However, the feedforward neural network is inflexible and unable to take inputs of varying sizes, the input layer must have as many units as there are nucleotides in the sequence. Consequently, only nucleotide sequences of the same length can have predictions rendered.

With test accuracies of 98% and 74.94% respectively, the RSGM algorithm and the feedforward neural network both outperform the LSTM mode (See Fig. 4.4). However, as described the feedforward neural network and RSGM are inflexible and impractical for real-world applications. The neural network must be changed if there is any variation in sequence length and the RGSM algorithm cannot perform its training or predictions if any of the nucleotide sequences vary in length.

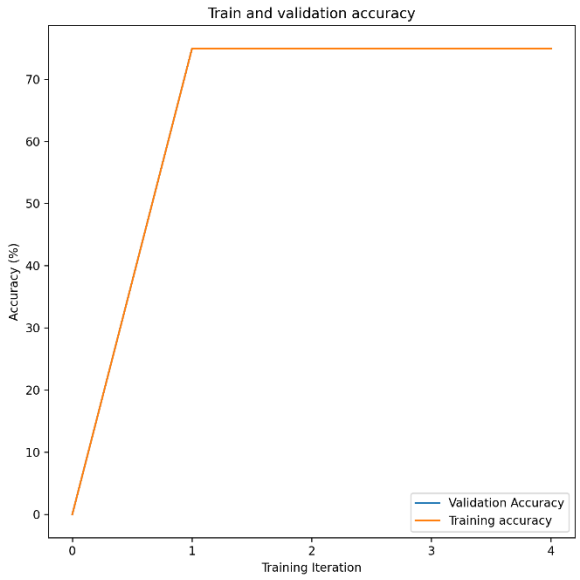


Figure 4.3 Feedforward neural network training and validation accuracy. Model – 1 hidden layer same width as nucleotide sequence length (ReLU), SoftMax, Learning rate – 0.5.

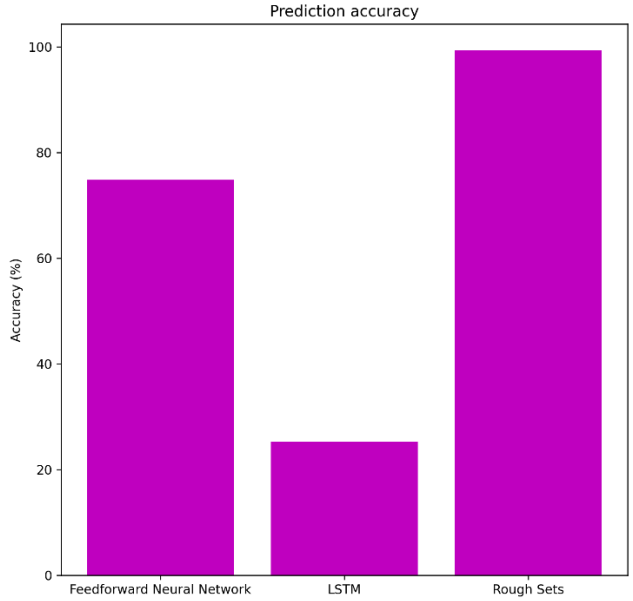


Figure 4.4 Prediction accuracy metrics for each model on test data. For the feedforward model accuracy metric were obtained through the Keras evaluation function which directly compares the input and target sequences and returns which proportion of predicted sequence is the same as the target sequence. With both the LSTM model and Rough Sets a nucleotide by nucleotide comparison must be done manually using the predicted output files.

With potentially zoonotic viruses, nucleotide sequences of the same length would be scarce. If the virus has never crossed the species barrier, the RSGM algorithm and feedforward neural network would be unviable because it requires at least two generations worth of nucleotide sequences of the same length to train. On the other hand, with its ability to adjust to varying nucleotide sequence length, the LSTM model can use nucleotide sequences of viruses closely related to the test virus for training and generate mutation predictions.

Given the most recent pandemic caused by a virus (COVID-19) that crossed the species barrier with relatively little known about its previous generations' genetics. The ability to predict its possible mutations using the genetic data from viruses closely related to it could have proved useful, because the predictions can be used to model on possible changes in infectiousness and lethality with this information being used to guide the initial response to the virus.

Chapter 5

5 Conclusion

In conclusion this project originally aimed to predict mutations that might cause a virus to become zoonotic (cross the species barrier into humans). Using a feedforward neural network and RSGM algorithm as baselines to compare against the novel LSTM model. The baseline models were chosen because they have already been trained, validated and tested as viral mutation predictors (Salama et al., 2016). Viral data was obtained from UVRI and wrangled into aligned, non-spaced nucleotide sequences before it was used to train, validate and test the models.

In conclusion the RSGM algorithm and the feedforward neural network produce more accurate results than the LSTM model. However, they are inflexible, and thus cannot be implemented in most real-world applications. The LSTM model, though less accurate, is better fitted to the training data and more versatile. Ideally nucleotide sequences as triplet codons would be the more ideal input but require larger amounts of training data if validation and test accuracies are to be improved.

6 Future Directions

With the continuous advances in computing power, NLP and RNN's, analysis of longer nucleotide sequences will be possible and not restricted to individual proteins due to a lack of processing power required by current models. This would enable predictions across the entire genome of a virus and possibly reveal, previously unknown, interactions between genetic positions. Furthermore, as more nucleotide sequences become available, consequently increasing the size of the training dataset, the accuracy of model predictions will increase. With better prediction accuracy a classifier can be added to the model which could determine whether predicted nucleotide sequences can cause zoonosis in a virus.

Misfolded proteins are central in modern neuroscience as they hold a pivotal role in the progression of many dementias. If prediction of viral genetic proves successful, the model can be added to the work currently being done to predict protein folding (Senior et al., 2020), and with this advancement possibly understand why proteins misfold, giving potentially vital insight into dementia.

Chapter 6

7 Bibliography

Viralzone.expasy.org. 2020. *Ebolavirus Cycle ~ Viralzone Page*. [online] Available at: <https://viralzone.expasy.org/5016> [Accessed 29 April 2020].

Agostinelli, F., Hoffman, M., Sadowski, P. and Baldi, P., 2014. Learning activation functions to improve deep neural networks. *arXiv preprint arXiv:1412.6830*.

Asgari, E. and Mofrad, M., 2015. Continuous Distributed Representation of Biological Sequences for Deep Proteomics and Genomics. *PLOS ONE*, [online] 10(11), p.e0141287. Available at: <https://www.ncbi.nlm.nih.gov/pmc/articles/PMC4640716/> [Accessed 12 May 2020].

Bamford, D., Burnett, R. and Stuart, D., 2002. Evolution of Viral Structure. *Theoretical Population Biology*, [online] 61(4), pp.461-470. Available at: <https://www.sciencedirect.com/science/article/pii/S0040580902915911> [Accessed 28 April 2020].

Barash, D. and Churkin, A., 2010. Mutational analysis in RNAs: comparing programs for RNA deleterious mutation prediction. *Briefings in Bioinformatics*, [online] 12(2), pp.104-114. Available at: <https://academic.oup.com/bib/article/12/2/104/261035> [Accessed 7 May 2020].

Beijerinck, M.W., 1898. On a Contagium vivum fluidum causing the Spotdisease of the Tobacco-leaves. *Koninklijke Nederlandse Akademie van Wetenschappen Proceedings Series B Physical Sciences*, *1*, pp.170-176.

Bonikowski, Z., Bryniarski, E. and Wybraniec-Skardowska, U., 1998. Extensions and intentions in the rough set theory. *Information Sciences*, [online] 107(1-4), pp.149-167. Available at: <https://www.sciencedirect.com/science/article/pii/S0020025597100469> [Accessed 28 April 2020].

Byvatov, E. and Schneider, G., 2003. Support vector machine applications in bioinformatics. *Applied bioinformatics*, *2*(2), pp.67-77.

Cao, W., Wang, X., Ming, Z. and Gao, J., 2018. A review on neural networks with random weights. *Neurocomputing*, [online] 275, pp.278-287. Available at: <https://www.sciencedirect.com/science/article/pii/S0925231217314613?casa_token=dIImW8TkZSkAAAAA:TJSYnR4BEZf2pMkXyK-nHf_7kkiKoaygdGlBqurQUOUbrvOcTX4810csig61fqSc-xt8ju1j7w> [Accessed 29 April 2020].

Cauthron, M., 2020. *Rough Sets Tutorial. - Ppt Download*. [online] Slideplayer.com. Available at: <https://slideplayer.com/slide/3236362/> [Accessed 13 May 2020].

Colah.github.io. 2020. *Understanding LSTM Networks -- Colah's Blog*. [online] Available at: <http://colah.github.io/posts/2015-08-Understanding-LSTMs/> [Accessed 18 September 2020].

Feldmann, H., 2014. Ebola — A Growing Threat?. *New England Journal of Medicine*, [online] 371(15), pp.1375-1378. Available at: <https://www.nejm.org/doi/full/10.1056/nejmp1405314> [Accessed 28 April 2020].

Fisher, R., 1999. *The Genetical Theory Of Natural Selection*. 1st ed. Oxford: Oxford University Press, pp.120-124.

Gehrmann, S., Deng, Y. and Rush, A.M., 2018. Bottom-up abstractive summarization. arXiv preprint arXiv:1808.10792.

Gelman, Andrew; Carlin, John B.; Stern, Hal S.; Dunson, David B.; Vehtari, Aki; Rubin, Donald B. (2013). Bayesian Data Analysis, Third Edition. Chapman and Hall/CRC. ISBN 978-1-4398-4095-5 (pp. 1-20).

Glorot, X. and Bengio, Y., 2010, March. Understanding the difficulty of training deep feedforward neural networks. In Proceedings of the thirteenth international conference on artificial intelligence and statistics (pp. 249-256).

Gulcehre, C., Moczulski, M., Denil, M. and Bengio, Y., 2016, June. Noisy activation functions. In International conference on machine learning (pp. 3059-3068).

Goldsmith, C. and Miller, S., 2009. Modern Uses of Electron Microscopy for Detection of Viruses. *Clinical Microbiology Reviews*, [online] 22(4), pp.552-563. Available at: <https://www.ncbi.nlm.nih.gov/pmc/articles/PMC2772359/> [Accessed 27 April 2020].

Gulcehre, C., Firat, O., Xu, K., Cho, K., Barrault, L., Lin, H. C., ... & Bengio, Y. (2015). On using monolingual corpora in neural machine translation. arXiv preprint arXiv:1503.03535.

Han, J., Kamber, M. and Pei, J., 2012. *Data Mining*. 4th ed. Amsterdam: Elsevier/Morgan Kaufmann, pp.393 - 442.

Hofacker, I., Fekete, M. and Stadler, P., 2002. Secondary Structure Prediction for Aligned RNA Sequences. *Journal of Molecular Biology*, [online] 319(5), pp.1059-1066. Available at: <https://www.sciencedirect.com/science/article/abs/pii/S002228360200308X?via%3Dihub> [Accessed 7 May 2020].

Houser, K. and Subbarao, K., 2015. Influenza Vaccines: Challenges and Solutions. *Cell Host & Microbe*, [online] 17(3), pp.295-300. Available at: <http://10.1016/j.chom.2015.02.012> [Accessed 27 April 2020].

Isken, O., 2004. Complex signals in the genomic 3' nontranslated region of bovine viral diarrhea virus coordinate translation and replication of the viral RNA. *RNA*, [online] 10(10), pp.1637-1652. Available at: <https://rnajournal.cshlp.org/content/10/10/1637.short> [Accessed 28 April 2020].

Klein, G., Kim, Y., Deng, Y., Senellart, J. and Rush, A., 2017. OpenNMT: Open-Source Toolkit for Neural Machine Translation. *Proceedings of ACL 2017, System Demonstrations*, [online] Available at: <https://opennmt.net/OpenNMT-py/> [Accessed 21 September 2020].

Lai, M., 1992. Genetic Recombination in RNA Viruses. *Current Topics in Microbiology and Immunology*, [online] 176, pp.21-32. Available at: <https://link.springer.com/chapter/10.1007/978-3-642-77011-1_2#citeas> [Accessed 29 April 2020].

Le Calvez, H., Yu, M. and Fang, F., 2004. *Virology Journal*, [online] 1(1), p.12. Available at: <https://www.ncbi.nlm.nih.gov/pmc/articles/PMC535550/> [Accessed 27 April 2020].

LeCun, Y., Bengio, Y. and Hinton, G., 2015. Deep learning. *Nature*, [online] 521(7553), pp.436-444. Available at: <https://www.nature.com/articles/nature14539> [Accessed 10 May 2020].

Lee, J. and Saphire, E., 2009. Ebola virus glycoprotein structure and mechanism of entry. *Future Virology*, [online] 4(6), pp.621-635. Available at: <https://www.ncbi.nlm.nih.gov/pmc/articles/PMC2829775/> [Accessed 27 April 2020].

Morin, R., Bainbridge, M., Fejes, A., Hirst, M., Krzywinski, M., Pugh, T., McDonald, H., Varhol, R., Jones, S. and Marra, M., 2020. Profiling The Hela S3 Transcriptome Using Randomly Primed Cdna And Massively Parallel Short-Read Sequencing. [online] Future Science. Available at: <https://www.future-science.com/doi/10.2144/000112900> [Accessed 7 May 2020].

Orengo, C., Todd, A. and Thornton, J., 1999. From protein structure to function. *Current Opinion in Structural Biology*, [online] 9(3), pp.374-382. Available at: <https://www.sciencedirect.com/science/article/pii/S0959440X99800517?casa_token=CodPG9EbKAQAAAAA:2sexxRpVKkCbsl7FAJir2Ri4yWaFcUFcZp6SSnKiXhQo8FYHD64HhhulPjcH3s8u-Jx0kw7t-A> [Accessed 29 April 2020].

Pawlak, Z., 1998. Rough Set Theory And Its Applications To Data Analysis. *Cybernetics and Systems*, [online] 29(7), pp.661-688. Available at: <https://www.tandfonline.com/doi/abs/10.1080/019697298125470?casa_token=U2kdRV-mUeEAAAAA:7NHTq696MyhQSXZIJG1Nt616UoldvXT_jdy2CFZShIRqxlT7E-JKJ2nUytJmxI8fi4sVBuNFVyX3> [Accessed 28 April 2020].

Ramachandran, P., Zoph, B. and Le, Q.V., 2017. Searching for activation functions. *arXiv preprint arXiv:1710.05941*.

Sanger, F., Nicklen, S. and Coulson, A., 1977. DNA sequencing with chain-terminating inhibitors. *Proceedings of the National Academy of Sciences*, [online] 74(12), pp.5463-5467. Available at: <https://www.ncbi.nlm.nih.gov/pmc/articles/PMC431765/> [Accessed 16 September 2020].

Sanjuán, R. and Domingo-Calap, P., 2016. Mechanisms of viral mutation. *Cellular*

and Molecular Life Sciences, [online] 73(23), pp.4433-4448. Available at: <https://www.ncbi.nlm.nih.gov/pmc/articles/PMC5075021/> [Accessed 27 April 2020].

Saunders-Hastings, P. and Krewski, D., 2016. Reviewing the History of Pandemic Influenza: Understanding Patterns of Emergence and Transmission. Pathogens, [online] 5(4), p.66. Available at: <https://www.ncbi.nlm.nih.gov/pmc/articles/PMC5198166/> [Accessed 12 April 2020].

Senior, A., Evans, R., Jumper, J., Kirkpatrick, J., Sifre, L., Green, T., Qin, C., Žídek, A., Nelson, A., Bridgland, A., Penedones, H., Petersen, S., Simonyan, K., Crossan, S., Kohli, P., Jones, D., Silver, D., Kavukcuoglu, K. and Hassabis, D., 2020. Improved protein structure prediction using potentials from deep learning. *Nature*, [online] 577(7792), pp.706-710. Available at: <https://www.nature.com/articles/s41586-019-1923-7> [Accessed 23 September 2020].

Sharp, P. and Hahn, B., 2011. Origins of HIV and the AIDS Pandemic. *Cold Spring Harbor Perspectives in Medicine*, [online] 1(1), pp.a006841-a006841. Available at: <https://www.ncbi.nlm.nih.gov/pmc/articles/PMC3234451/> [Accessed 27 April 2020].

Shterionovα, D., Casanellasβ, P.N.L., Superboβ, R. and O'Dowdβ, T., Empirical evaluation of NMT and PBSMT quality for large-scale translation production. In *Conference Booklet* (p. 74).

Smith, L., 2017. Cyclical Learning Rates for Training Neural Networks. *2017 IEEE Winter Conference on Applications of Computer Vision (WACV)*, [online] Available at: <https://ieeexplore.ieee.org/abstract/document/7926641> [Accessed 30 April 2020].

Stawicki, S., Arquilla, B., Galwankar, S., Hoey, B., Jahre, J., Kalra, S., Kelkar, D., Papadimos, T., Sabol, D. and Sharpe, R., 2014. The emergence of Ebola as a global health security threat: From ′lessons learned′ to coordinated multilateral containment efforts. Journal of Global Infectious Diseases, [online] 6(4), p.164. Available at: <https://www.ncbi.nlm.nih.gov/pmc/articles/PMC4265832/> [Accessed 12 April 2020].

Tch, A., 2020. *The Mostly Complete Chart Of Neural Networks, Explained*. [online] Medium. Available at: <https://towardsdatascience.com/the-mostly-complete-chart-of-neural-networks-explained-3fb6f2367464> [Accessed 29 April 2020].

Van Noord, R. and Bos, J., 2017. Neural semantic parsing by character-based translation: Experiments with abstract meaning representations. arXiv preprint arXiv:1705.09980.

Walker, J., Han, B., Ott, I. and Drake, J., 2018. Transmissibility of emerging viral zoonoses. *PLOS ONE*, [online] 13(11), p.e0206926. Available at: <https://journals.plos.org/plosone/article?id=10.1371/journal.pone.0206926> [Accessed 27 April 2020].

World Health Organization. 2020. *Zoonoses*. [online] Available at: <https://www.who.int/topics/zoonoses/en/> [Accessed 10 May 2020].

Wu, Y., Schuster, M., Chen, Z., Le, Q. V., Norouzi, M., Macherey, W., ... & Klingner, J. (2016). Google's Neural Machine Translation System: Bridging the Gap between Human and Machine Translation. arXiv preprint arXiv:1609.08144.

Yin, R., Luusua, E., Dabrowski, J., Zhang, Y. and Kwoh, C., 2020. Tempel: time-series mutation prediction of influenza A viruses via attention-based recurrent neural networks. *Bioinformatics*, [online] 36(9), pp.2697-2704. Available at: <https://academic.oup.com/bioinformatics/article/36/9/2697/5717964> [Accessed 12 May 2020].

Yu, D., Weng, T., Wu, X., Wang, F., Lu, X., Wu, H., Wu, N., Li, L. and Yao, H., 2017. The lifecycle of the Ebola virus in host cells. *Oncotarget*, [online] 8(33), pp.55750–55759. Available at: <https://www.ncbi.nlm.nih.gov/pmc/articles/PMC5589696/> [Accessed 29 April 2020].

Zhang, Q., Yang, Y., Ma, H. and Wu, Y.N., 2019. Interpreting cnns via decision trees. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition* (pp. 6261-6270).

Zhong, N. and Zeng, G., 2006. What we have learnt from SARS epidemics in China. BMJ, [online] 333(7564), pp.389-391. Available at: <https://www.ncbi.nlm.nih.gov/pmc/articles/PMC1550436/> [Accessed 12 April 2020].