**Distribution Agreement**

In presenting this thesis or dissertation as a partial fulfillment of the requirements for an advanced degree from Emory University, I hereby grant to Emory University and its agents the non-exclusive license to archive, make accessible, and display my thesis or dissertation in whole or in part in all forms of media, now or hereafter known, including display on the world wide web. I understand that I may select some access restrictions as part of the online submission of this thesis or dissertation. I retain all ownership rights to the copyright of the thesis or dissertation. I also retain the right to use in future works (such as articles or books) all or part of this thesis or dissertation.

Signature:

| | |
|---|---|
| _____ | ____04/10/2020____ |
| Zhenghao Chu | Date |

# Using deep learning methods to predict the VRC01 neutralization sensitivity by HIV-1 gp160 sequence features

By

Zhenghao Chu
Master of Science in Public Health

Bioinformatics and Biostatistics

---

Committee Chair

David Benkeser, PhD

---

Committee Member

Hao Wu, PhD

# Using deep learning methods to predict the VRC01 neutralization sensitivity by HIV-1 gp160 sequence features

By

Zhenghao Chu

B.S.
Fudan University
2015

Thesis Committee Chair: David Benkeser, PhD

An abstract of
A thesis submitted to the Faculty of the
Rollins School of Public Health of Emory University
in partial fulfillment of the requirements for the degree of
Master of Science in Public Health
in Bioinformatics and Biostatistics
2020

## Abstract

Using deep learning methods to predict the VRC01 neutralization sensitivity by HIV-1 gp160 sequence features

By Zhenghao Chu

**Introduction:** The broadly neutralizing antibody (bnAb) VRC01 is being evaluated for its efficacy to prevent HIV-1 infection in the Antibody Mediated Prevention (AMP) trials. Our object is to applied Deep learning (DL) methods to see whether or not DL models can help improve accuracy on predicting sensitivity of neutralization of 611 HIV-1 Env pseudoviruses by VRC01.

**Methods:** We tried three different kinds of Deep Neural Network structures (FCNN, 1D-CNN, 1D-CNN+BiLSTM) to do the prediction and implemented a 5-fold cross-validation method to verify the performance of the model. We chose best model of each three neural network structures to do the prediction on our test set. We selected accuracy (Accuracy, Acc), precision (P), recall (R), F1 score (F1) and average area under the receiver operating characteristics (ROC) curve as evaluation indicators.

**Results:** The three mean AUCs (area under curve) for ROC curves are $0.857 \pm 0.070$, $0.763 \pm 0.076$, $0.755 \pm 0.075$ respectively. The prediction accuracies are 0.85, 0.83, 0.85 respectively.

**Conclusion:** For this small sample size task, our three Deep Learning models did not perform as well as the random forest model.



**Key words:** Human Immunodeficiency Virus (HIV), Antibody Mediated Prevention (AMP), Deep Learning, Deep Neural Network (DNN), Recurrent Neural Network (RNN), Convolutional Neural Network (CNN), Long Short-Term Memory (LSTM)

# Using deep learning methods to predict the VRC01 neutralization sensitivity by HIV-1 gp160 sequence features

By

Zhenghao Chu

B.S.
Fudan University
2015

Thesis Committee Chair:

A thesis submitted to the Faculty of the
Rollins School of Public Health of Emory University
in partial fulfillment of the requirements for the degree of
Master of Science in Public Health
in Bioinformatics and Biostatistics
2020

## Acknowledgement:

I would like to express my deep gratitude to Professor David Benkerser, my thesis advisor, and Professor Hao Wu, my thesis reader, for their patient guidance, enthusiastic encouragement and useful critiques of this research work. I would also like to thank Yunchuan Kong and YanTing Huang, my seniors, for their advice and assistance on my thesis.

I would also like to extend my thanks to the technicians of the laboratory of the Bioinformatics and Biostatistics department for their help in offering me the resources in running the program.

Finally, I wish to thank my parents for their support and encouragement throughout my study.

# Introduction

HIV (human immunodeficiency virus) remains a major public health issue. According to the statistics by WHO, more than 32 million people have died of AIDS since the first AIDS case reported in 1981 [1, 2] and 37.9 million people living with HIV [2] by 2018. There are two main types of HIV – HIV-1 (accounting for around 95% of all infections worldwide) and HIV-2 (relatively uncommon and less infectious) [3]. Although there is no cure for HIV now, great progress has been made on HIV prevention and treatment. Current guidelines [4] recommend pre-exposure prophylaxis (PrEP) as an effective prevention method for individuals at high risk of HIV infection [5]. PrEP regimens require taking a pill every day and adherence is required to maintain protective efficacy [6]. In some individuals, adhering to a strict PrEP regimen may be difficult. It is therefore also of interest to develop a prophylactic HIV vaccine [7]. Ongoing clinical trials are evaluating the efficacy and safety of HIV vaccines [8], but as of now, there is no licensed vaccine available to prevent HIV. Development of a vaccine is challenging owing to rapid mutations of the HIV pathogen and the difficulty of inducing a robust immune response to a preventive vaccine [9]. Nevertheless, the RV144 vaccine was observed to confer partial efficacy in a clinical trial in Thailand, reported in 2009.

A new approach to prevention involves the usage monoclonal antibodies. Researchers have isolated broadly neutralizing antibodies (bnAbs) from individuals with chronic HIV infection, which have shown the ability to neutralize a wide spectrum of HIV-1 viruses [10]. In 2016, two multinational clinical trials of an intravenously delivered monoclonal antibody for preventing HIV infection were launched; together these are known as AMP (antibody-mediated prevention) Studies. Participants were randomized to receive an investigational bnAb called VRC01, an antibody discovered in an HIV-infected person whose body was able to control the infection without using any antiretroviral drugs [11]. Researchers found that the major target of neutralizing antibodies is the trimeric HIV-1 envelope (Env) glycoprotein spike [precursor form = $(gp160)_3$, proteolytically cleaved to $(gp120/gp41)_3$]. Immense genetic and antigenic diversity of the envelope glycoprotein causes a significant problem in the development of an effective prophylactic vaccine [12, 13]. However, bnAbs generally target conserved elements of gp160 across five different regions: the V2 variable region, the N332 region in the V3 region, the CD4 binding site (CD4bs), the gp120–gp41 interface, and the membrane proximal external region [14]. It is therefore of interest to better understand the relationship between Env amino acid (AA) signatures and the neutralization phenotype of interest [15].

Due to the highly diversity of HIV-1's outer (Env) protein, a key secondary goal of the AMP Studies is to assess whether VRC01 confers differential efficacy based on amino acid features in the Env glycoprotein, using amino acid sequence sieve analysis [16-19]. To prepare for this analysis, we must a-priori identify amino acid features that are most likely relevant to neutralization sensitivity by VRC01. Previously, Magaret et al. used Super Learner, a nonparametric ensemble-based cross-validated learning method, to predict neutralization sensitivity using in vitro virus sequences obtained from the CATNAP database [20, 21]. The authors also compared this approach with other machine learning methods such as Random Forest, boosting and Lasso to examine the prediction of sensitivity and to rank AA features by their predictive importance. They found the Super Learner model performed reasonably well in classifying $IC_{50}$-based

{IC50 is a quantitative measure that indicates how much of a particular inhibitory substance (in our case VRC01) is needed to inhibit, in vitro, a given biological process or biological component by 50% [22]. The definition for sensitive is (IC50 < 1 μg/mL) and resistant (right-censored IC50 value)} dichotomous outcomes for VRC01 neutralization with an average validated AUC of 0.868 across two hold-out datasets. The study also identified the top 50 features including 26 surface-accessible residues in the VRC01 and CD4 binding footprints, length of gp120 protein, the length of Env protein, the number of cysteines in gp120, the number of cysteines in Env, and 4 potential N-linked glycosylation sites [23].

Accurate prediction of neutralization sensitivity is likely to become an increasingly important scientific goal, with further AMP studies already in development. It is therefore of interest to provide a thorough examination of machine learning methods for predicting neutralization sensitivity. A notable omission from the work of Magaret et al was the usage of deep learning approaches. Deep learning (DL) algorithms have been successfully applied in many different fields including image recognition, natural language and voice processing. Researchers have also used DL to help elucidate biological structures in the context of genomics [24]. In particular, Alipanahi et al. developed an DL algorithm which can help them predict the sequence specificities of DNA- and RNA-binding proteins [25]. Due to the similarity of primary structure of DNA/RNA (generally consisting of 4 types of nucleotides) and proteins (generally consisting of 20 types of amino acids), it is of interest to examine whether similar DL approaches can help improve prediction of neutralization sensitivity in the context of HIV. In this work, we provide the first exploration of this question, applying deep learning methods to prediction of neutralization of 611 HIV-1 Env pseudoviruses by VRC01 using the CATNAP database. We examine how network architecture may affect performance in this setting and compare the performance of DL to other machine learning approaches.

# Method

## 1. Objective

The objective of our work is to develop three deep learning models for classifying TZM-bl neutralization sensitivity to VRC01. We constructed three deep neural network structures (1) a fully-connected neural network (FCNN); (2) An one dimensional convolutional neural network (1D-CNN); (3) One dimensional convolutional neural network + Bidirectional Long short-term memory recurrent neural network (1D-CNN + BiLSTM) to conduct this classification, all of which used a set of pre-defined AA sequence features to predict TZM-bl [26] neutralization outcomes indicating whether a virus is right-censored/resistant to VRC01 (defined as having $IC_{50} > 10$ μg/ml[21]).

## 2. Dataset

A total of 611 sequences/pseudoviruses were included in this analysis and their associated pseudovirus values for neutralization by VRC01 as assessed by the TZM-bl assay [26], and other associated annotations were downloaded from the CATNAP database [20]. We randomly separated our dataset into two datasets ("dataset 1" and "dataset 2") for the statistical learning analyses. The two datasets were mutually exclusive, each with half of the data [n = 306 (dataset 1) and n = 305 (dataset 2)]. More information about dataset can be found in Magaret's paper[23].

### 2.1 Input variables

We separated our input variables into two groups: 1. Envelope amino acid (AA) position features (what kind of amino acid located on different important AA positions of Envelope proteins). 2. Non-amino acid (Non-AA) position features including: Subtype of HIV-1 virus, Geographic information (region of origin), Indication of potential N-linked glycosylation sites (PNGS), Region-specific counts of PNG sites, Viral geometry, Cysteine counts, Steric bulk at critical locations.

### 2.2 Output variable

Pseudoviruses whose $IC_{50}$ was right-censored were labeled "resistant" to neutralization by VRC01; the rest of the virsuses were labeled "sensitive" to neutralization. We used all the input variables with different preprocessing methods in our models and use them to predict our outcome and evaluate our models.

## 3. Perceptron

Deep neural network (DNN) is an information processing paradigm inspired by the way biological systems such as the brain process information. It integrates of a large number of highly interconnected processing elements (neurons) to provide a prediction of an outcome. The basic structure of DNN is called a perceptron (Fig1).

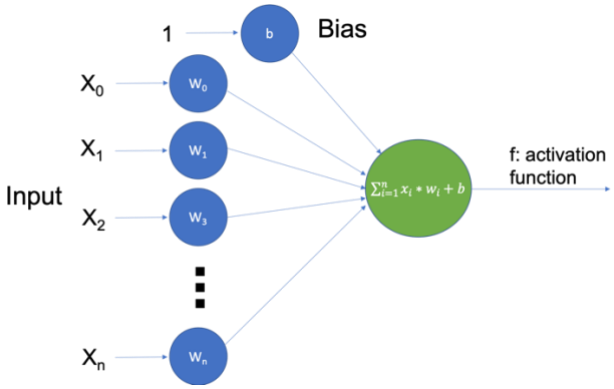



**Fig1. Perceptron**

Suppose we have n genetic and phenotypic sequence features measured on a given pseudovirus. Each feature is multiplied by a connection weight and then add an intercept b to them (Fig1). The weights are denoted by $w_0$, $w_1$, $w_2$, $w_3$, …, $w_n$. The value of each weight indicates the strength of a particular node in "activating" the neuron. The function "**f**" is called the activation function and is a non-linear function that determines whether a neuron should be activated or not by calculating the linear combination of features. Different kinds of activation functions can be used. In regression problems, it is common to use the identity function, $output = \sum_{i=1}^{n} x_i * w_i + b$ , while in classification problems, sigmoid or softmax activation functions are commonly used.

## 4. Fully Connected Neural Network (FCNN)

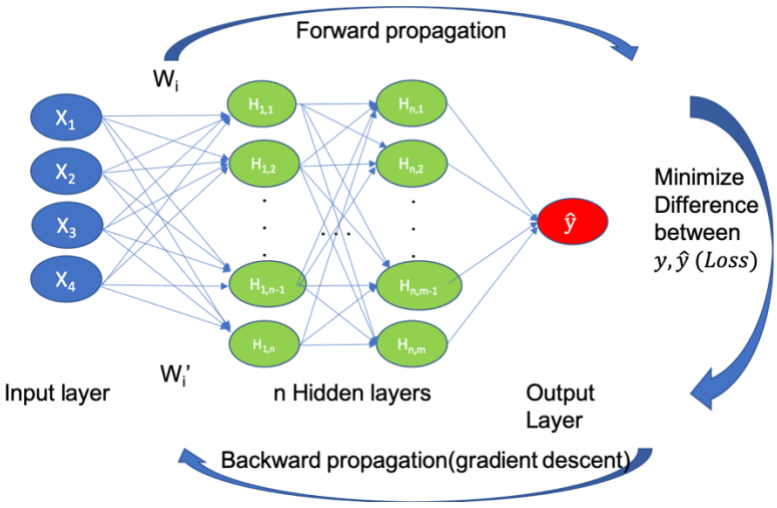



**Fig2. Fully connected neural network and training procedure**

A fully connected neural network (FCNN) is also known as multiple layer perceptron (MLP). A FCNN combines many perceptrons together and to form so-called hidden layers. Fig2 illustrates this idea for four input features and two hidden layers, each with n neurons. Each neuron in the first hidden layer is a perceptron, as described above. Then, each neuron in the second hidden layer uses the neurons from the first hidden layer as input variables, and

creates another perceptron. In this structure, we say that neurons are *fully connected.*

Once we decide our neural network structures and all activation functions, we can then use our data to train our neural network. The training aims at minimize the difference between the real output values and predicted values by changing the weights.

The training procedure are as follows:
1. Randomly initialize all weights to a decimal close to but not equal to zero.
2. Input the first observation of the dataset in the input layer, with each feature in a node.
3. Forward propagation: Neurons are activated from left to right in a way that the influence of each neuron activation is limited by weight. The propagation is activated until the predicted value is obtained.
4. Compare the predicted results with the actual results and calculate the prediction error

$$\mathrm{J(W,b)} = \frac{1}{n}\sum_{i=1}^{n} l(f(X_i; W, b), y_i)$$

where W is matrix of all weights; $X_i$ is input features; $y_i$ is real label;
$f(X_i; W, b) = \widehat{y_i}$
$l(f(X_i; W, b), y_i)$ is a loss function evaluated on each sample. For example, the loss could be the square of each sample's residual.

$$l(f(X_i; W, b), y_i) = (f(X_i; W, b) - y_i)^2$$

5. Back Propagation: From right to left, errors *propagate back*, that is, based on predictive performance for the current value of weights, new values are assigned using a strategy based on gradient descent. A learning rate determines the magnitude of the weight update. At the step t+1, weights are updated according as

$$W_{t+1} = W_t - \eta_w * \nabla \mathrm{w} L(W_t)$$
$$b_{t+1} = b_t - \eta_b * \nabla \mathrm{b} L(b_t)$$

where $W_t$ is the current weight matrix, $W_{t+1}$is the weight matrix after gradient descent, $\eta$ is learning rate and

$$\nabla \mathrm{w} = \frac{\partial \mathrm{J(W,b)}}{\partial W}$$
$$\nabla \mathrm{b} = \frac{\partial \mathrm{J(W,b)}}{\partial b}$$

are the gradients of the loss function.
6. Repeat steps 1 to 5 and update the weights after each observation. One cycle through the data set updating the weights is referred to as an *epoch*.
7. The process continues for a user-seleceted number of epochs or until some other stopping criteria is reached.

## 5. 1-Dimensional Convolutional Neural Network (1D-CNN)

Convolutional neural networks (CNN) [27] have proven to be effective for many natural language processing (NLP) tasks [28]. Inspired by their success in text

classification, in this paper, CNNs with various kernel sizes are used to extract local contexts from a protein sequence. In particular, we use a one-dimensional CNN, which kernelizes the architecture of the perceptron. That is, nodes in the perceptron are constructued based on features that are *near* to each other in some sense. Higher dimensional kernels are common in other contexts such as image recognition. CNN are useful for deriving features from a fixed-length segment of the overall dataset, where it is not so important where the feature is located in the segment.
These methods have been shown to predict DNA-binding proteins [25]. Delong et al. were the first to show that protein features can be identified by deep learning [25,29,30].
More detailed information on CNN algorithm can be found in the paper written by Murugan et al. [31].

## 6. 1-Dimensional Convolutional Neural Network + Bidirectional Long Short-Term Memory (1D-CNN+BiLSTM)

RNN (Recurrent Neural Network) [32] are designed to have an *internal memory*. While CNN only considers constructing neurons that include information from consecutive elements in a sequence, it may ignore relationships between non-continuous sequences. RNN incorporates information from more distant sequence elements. However, RNN often have difficulties associated with the gradient descent portion of training. BiLSTM (Bidirectional Long Short-Term Memory) [33] is an extension of RNN, which is specifically used to deal with this problem. These networks can more efficiently account for sequence information separated by long distances. By combining CNN and BiLSTM, not only can we account for local dependence between sequence features, but also more distal dependence.
More details of our implementation of all three types of DNN are included in Appendix.

## 7. Evaluation Criteria

We implemented a five-fold cross-validation scheme to compare the performance of the models. The training dataset (dataset 1) was divided into five parts. We take one part as the validation set and the other 4 parts as the training set. The networks are trained on the training set, while the performance is evaluated on the validation set. The process is repeated five times and performance is reported as the average performed over the five validations sets. Finally, we chose best model of each three neural network structures to do the prediction on our test data (dataset 2). We evaluated each model on several criteria and compared to predictions made by a random forest, which was found to have the strongest prediction on these data in previous work. We compared predictive performance in terms of accuracy (Acc), precision (P), recall (R), F1 score (F1) and area under the receiver operating characteristics (ROC) curve. The formulas are as follow:

$$Acc = \frac{TP + TN}{TP + TN + FP + FN}$$

$$P = \frac{TP}{TP + FP}$$

$$R = \frac{TP}{TP + FN}$$

$$F1 = \frac{2 * TP}{2 * TP + FN + FP}$$

| Real result | predicted result | |
|---|---|---|
| | Positive class | Negative class |
| Positive class | TP | FN |
| Negative class | FP | TN |

**Table1. meaning of classification results**

TP: true positive, FN: false negative, FP: false positive, TN: true negative

# Results

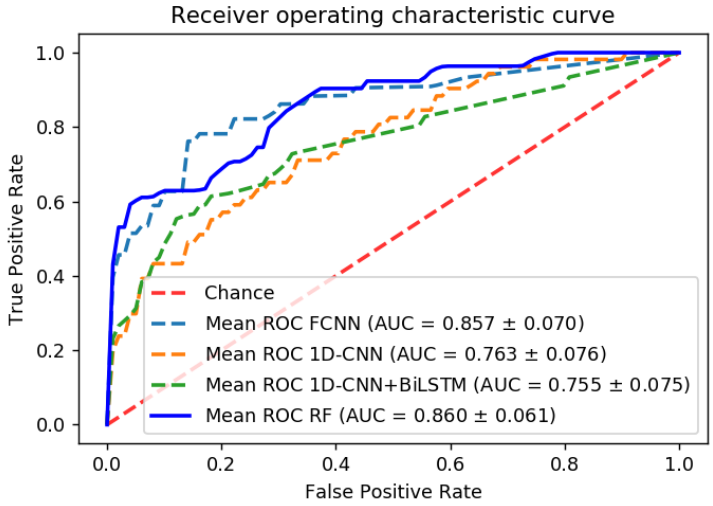



**Fig. 3 5-Fold Cross-validation ROC Curve for all three deep learning models and the Random Forest model**

The cross-validated AUC for 1D-CNN and 1D-CNN+BiLSTM was less than 0.8, while FCNN performed similarly to Random Forests (Fig 3).

| Model / Criteria | Accuracy | Precision | Recall | F-1 Score |
|---|---|---|---|---|
| FCNN | 0.85 | 0.9 | 0.47 | 0.91 |
| 1D-CNN | 0.83 | 0.88 | 0.34 | 0.9 |
| 1D-CNN + BiLSTM | 0.85 | 1 | 0 | 0.92 |
| Random Forest | 0.88 | 0.89 | 0.3 | 0.93 |

**Table 2. Evaluation Results on validation data**

The best models selected via five-fold cross-validation in dataset 1 were evaluated on dataset 2 (Table 2). All four models had high accuracy and the precision. However, the recall rate was low for most models and particularly low for 1D-CNN + BiLSTM. The FCNN yielded the highest recall rate. Random forest scored the highest in terms of F-1 score.

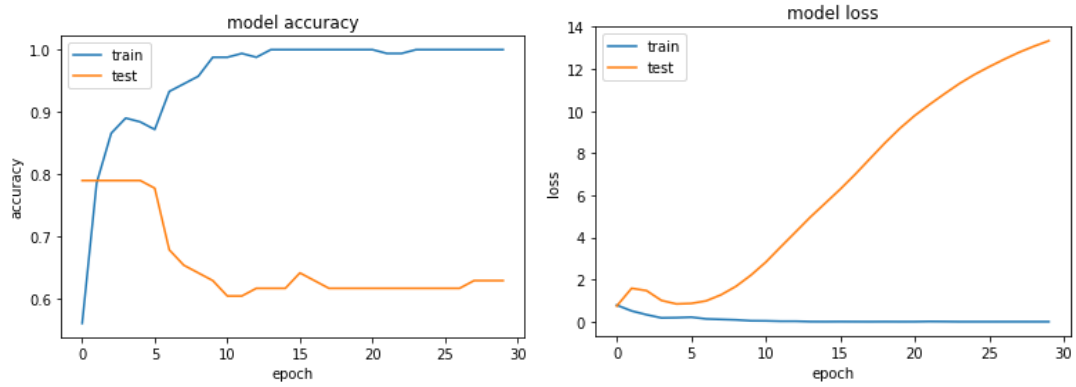



**Fig 4. Learning curve for FCNN model**

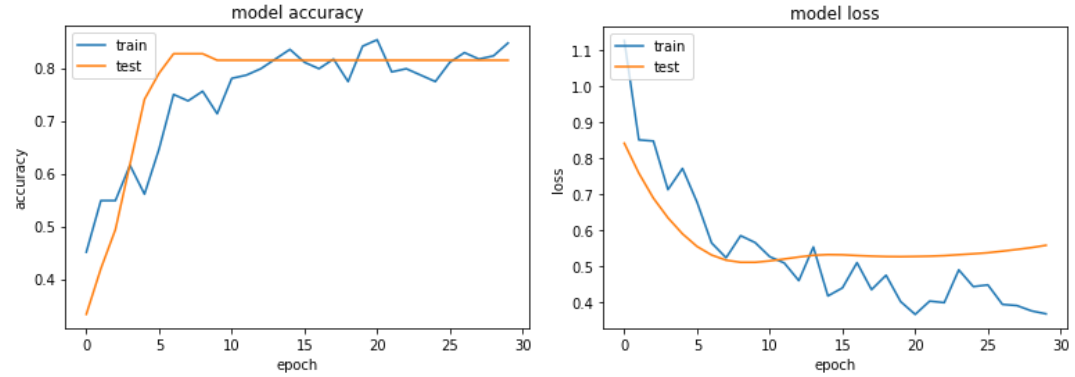



**Fig 5. Learning curve for 1D-CNN model**

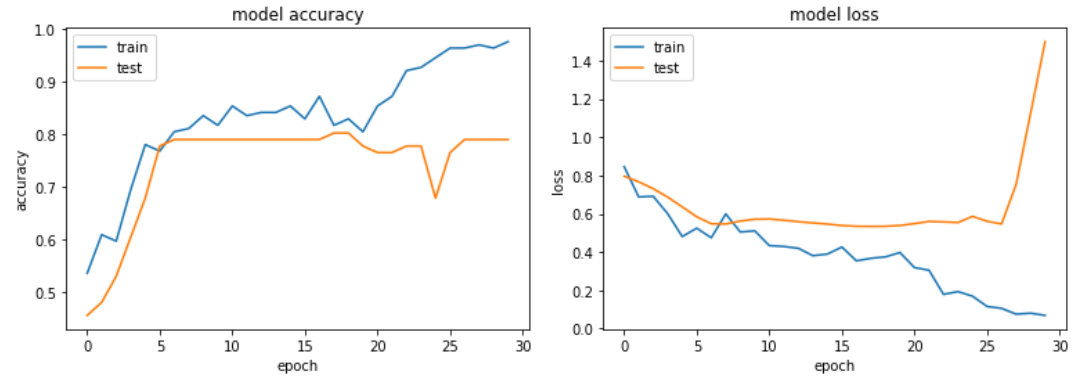



**Fig 6. Learning curve for 1D-CNN+BiLSTM model**

From Figs 4–6, we can see that the training curve and the validation curve of the 1D-CNN are closer than other two models, both for Acc and loss. This indicates that 1D-CNN experiences very little overfitting. Visibly, the training process of 1D-CNN and 1D-CNN + BiLSTM reflects the true performance of the data better than FCNN. In addition, FCNN will converge very quickly at the beginning of training, but will quickly reach the upper limit, and the val_loss value will explode after training for 5 cycles. In contrast, 1D-CNN and 1D-CNN + BiLSTM converged more slowly, initially showed a slow upward trend, and finally reached val_loss close to 0.1. Since 1D-CNN and 1D-CNN + BiLSTM use a more complex neural network than other models, it cannot quickly increase train_acc in the initial stage of training, but it will continuously reduce the loss value during continuous operation. Despite the longer training time, 1D-CNN and 1D-CNN + BiLSTM can capture amino acid sequence features in more detail than FCNN.

# Discussion

The low recall results highlight the issue of unbalanced data. Because most viruses are VRC01-sensitive, classification accuracy is very high for sensitive viruses but low for resistant viruses. There may be techniques that could improve performance in this challenging setting. For example, resampling technics are widely used to deal with imbalanced data. Chawla et al. used a synthetic minority over-sampling technique (SMOTE) [34] to up-sample the minority classes, which led to improved model performance. Another approach is to choose an alternative loss function for training the models. Lin et al.[35] developed a new type of loss function for imbalanced dataset called Focal Loss Function, which can perform better than binary cross-entropy loss in imbalanced data sets.

Another potential challenge of applying deep learning to these sequence data is that the sample size is relatively small. Traditional applications of deep learning generally involve training networks on very large data sets. In this example, we only have 306 for our training data, which maybe too limited for training our complicated neural networks. Nevertheless, the FCNN was able to achieve performance comparable to random forests in spite of these difficulties, indicating that future research into deep learning for neutralization prediction may be warranted.

For future plan, we will to dive into more details on weights of our models. With weights information, we can have a basic idea on which amino acid features are more important for prediction. Although the methods for doing feature importance for a deep learning model is rare, there are still some useful tools to achieve our goal such as variance-based feature importance, permutation importance and early stabilizing feature importance, etc.

# Appendix

## Network Architecture

All three neural network structures share the same type of hidden layer activation function, dropout layer, output layer activation function, loss function and optimizer, so for the same parts we only described it in our Fully Connected Neural Network below. For the different parts of three neural network structures (like input layer), we mention

### 1. Fully Connected Neural Network

#### 1.1 Input Layers

For non-amino-acid (Non-AA) position features, changed all the categorical features (Subtype of HIV-1 virus & Geographic information) into dummy variables. For amino acid (AA) position features, we also changed it into dummy variables (This coding method is also used by Magaret et al. [23]). See an example in Fig 7. Then we standardized all input variables to improve the performance of our models. By applying this kind of input variable coding, we simply focus on the information of whether or not different amino acid types of the same position can help predict the results.

**Fig. 7 AA position features preprocessing for FCNN using dummy coding.**

| | AA Position feature | |
|---|---|---|
| Sequence /sample # | hxb2.97 | hxb2.124 |
| 1 | E | P |
| 2 | K | F |
| 3 | N | F |
| 4 | E | P |

| | AA Position Feature | | | | |
|---|---|---|---|---|---|
| Sequence /sample # | hxb2.97.E | hxb2.97.K | hxb2.97.N | hxb2.124.P | hxb2.124.F |
| 1 | 1 | 0 | 0 | 1 | 0 |
| 2 | 0 | 1 | 0 | 0 | 1 |
| 3 | 0 | 0 | 1 | 0 | 1 |
| 4 | 1 | 0 | 0 | 1 | 0 |

In this example, we have 4 sample sequence and 2 AA position features hxb2.97 and hxb2.124. In position 97, we can find 3 types of AA ("E", "K", "N") and 2 types ("P", "F") in 124. ("E","K","N","P","F" are shorts for type of different amino acids.) There are total 22 types of amino acids in our dataset ("A","R","N","D","C","Q","E","G","H","I","L","K","M","F","P","S","T","W","Y","X","V","gap").

#### 1.2 Hidden Layers

We used 4 fully connected dense layers and each layer has different number of units (256, 64, 32 and 32 successively). Although there are many types of activation functions, Rectified Linear Units (ReLu) is the most used activation function in hidden layers of deep neural networks.

$$f(x) = x^{+} = \max(x, 0)$$

#### 1.3 Dropout Layer

We added an Alpha Dropout Layer [36] to prevent overfitting. The dropout rate was set to 0.5.

#### 1.4 Output Layer

We chose sigmoid function as our output layer activation function because it is a binary outcome classification problem.

$$Sigmoid(x) = \frac{1}{1 + e^{-x}}$$

### 1.5 Loss Function

Because our result is a binary outcome, we chose binary cross entropy as our loss function which performs better for classification models than mean square error (MSE).

$$loss = -\sum_{i=1}^{n} \hat{y_i} \log y_i + (1 - \hat{y_i}) \log (1 - y_i)$$

### 1.6 Optimizer

We chose Adam as our optimizer. Adam (adaptive moment estimation) is an adaptive learning rate method. It computes individual learning rates for different parameters. Adam uses estimations of first and second moments of gradient to adapt the learning rate for each weight of the neural network. Step size of Adam update rule is invariant to the magnitude of the gradient, which helps a lot when going through areas with tiny gradients [37].

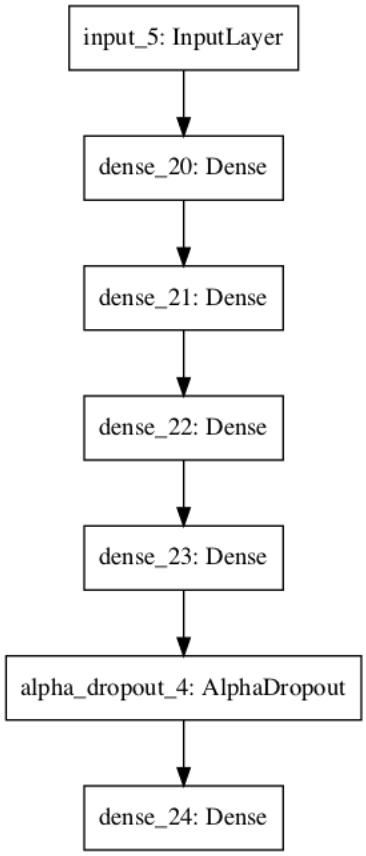


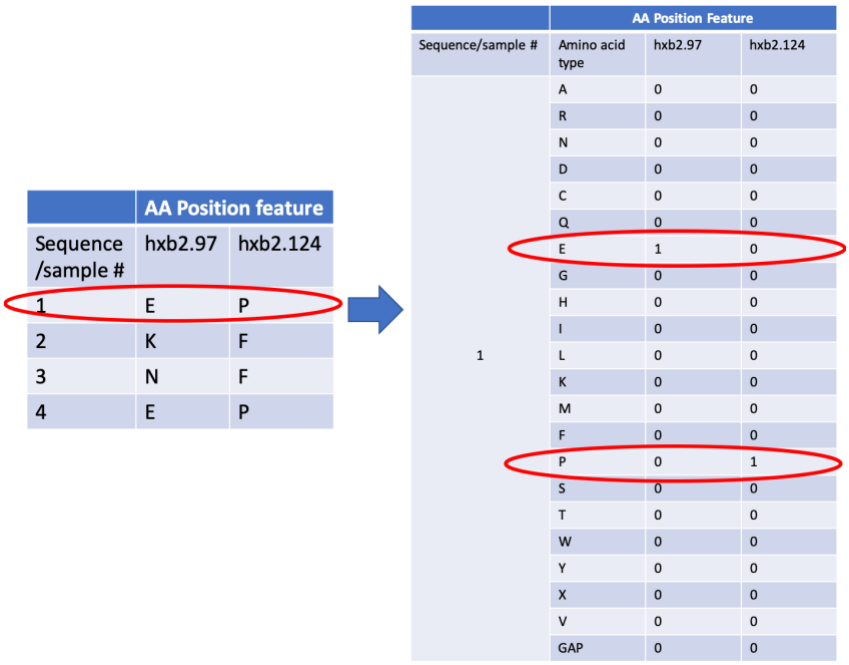

**Fig 8. Flow plot for FCNN**

## 2. 1D-CNN

### 2.1 Input Layer

For non-amino-acid (Non-AA) position features, changed all the categorical features (Subtype of HIV-1 virus & Geographic information) into dummy variables. For amino acid (AA) position features, we turned its AA position features into a 22 * 91 matrix (total number of AA type in our cases * total number of preselected AA position) for each sample using one hot encoding method. See an example in Fig 9. After that we also standardized all variables for the same purpose.

Using one hot encoding methods, we now focus on the information of whether or not certain amino acid position is important.



| | AA Position feature | |
|---|---|---|
| Sequence /sample # | hxb2.97 | hxb2.124 |
| 1 | E | P |
| 2 | K | F |
| 3 | N | F |
| 4 | E | P |

| | AA Position Feature | | |
|---|---|---|---|
| Sequence/sample # | Amino acid type | hxb2.97 | hxb2.124 |
| 1 | A | 0 | 0 |
| | R | 0 | 0 |
| | N | 0 | 0 |
| | D | 0 | 0 |
| | C | 0 | 0 |
| | Q | 0 | 0 |
| | E | 1 | 0 |
| | G | 0 | 0 |
| | H | 0 | 0 |
| | I | 0 | 0 |
| | L | 0 | 0 |
| | K | 0 | 0 |
| | M | 0 | 0 |
| | F | 0 | 0 |
| | P | 0 | 1 |
| | S | 0 | 0 |
| | T | 0 | 0 |
| | W | 0 | 0 |
| | Y | 0 | 0 |
| | X | 0 | 0 |
| | V | 0 | 0 |
| | GAP | 0 | 0 |

**Fig. 9 AA position features preprocessing for 1D-CNN using One hot encoding.**

In this example, we have 4 samples/sequences and 2 AA position features hxb2.97 and hxb2.124. For sample 1, we change the AA position features into a 22*2 matrix.

### 2.2 Batch Normalization Layer

Batch normalization is a technique for improving the speed, performance, and stability of deep neural networks. Batch normalization was introduced in a 2015 paper [38]. It is used to normalize the input layer by adjusting and scaling the activations.

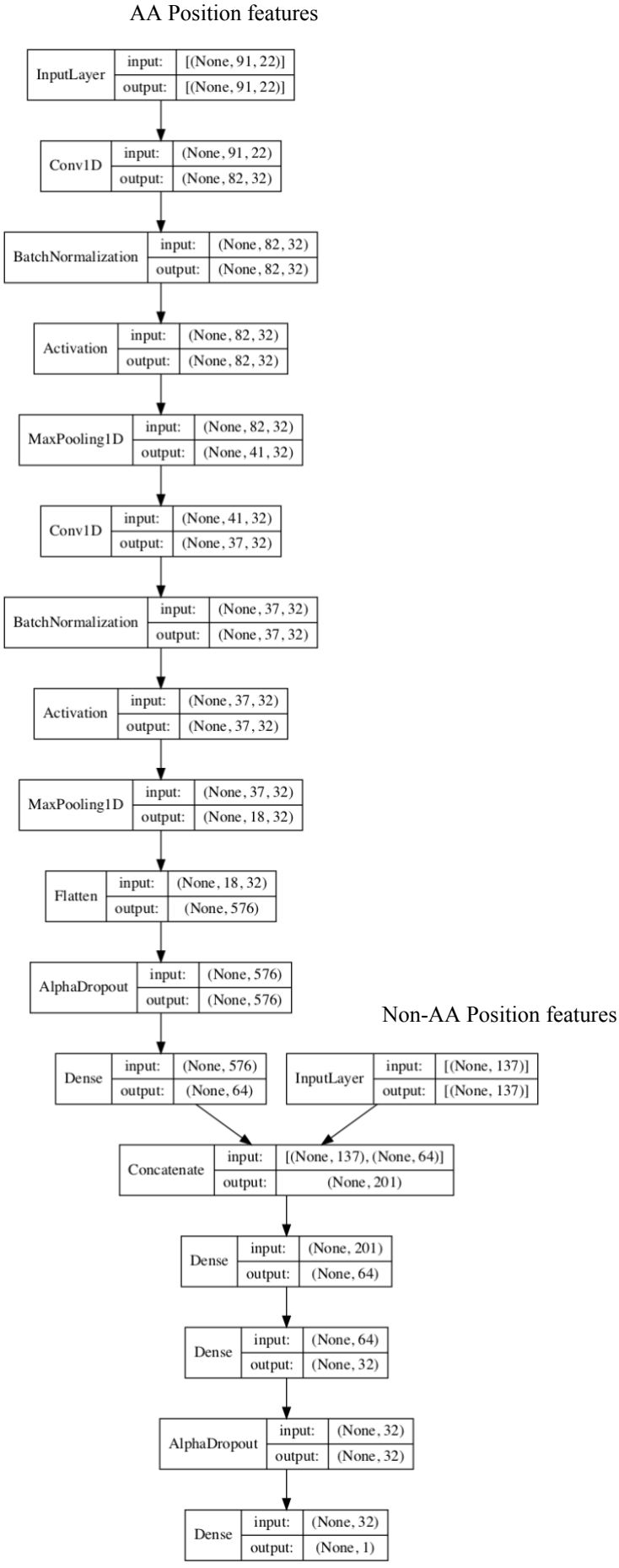



**Fig. 10 Flow plot for 1D-CNN**

# 3. 1D-CNN+BiLSTM

## 3.1 Input Layer

There are 22 different types of amino acids in our datasets. Each amino acid is represented by a capital letter (For sake of convenience, we set "gap" as one of the 22 types) [39]. We use different numbers to represent different types of amino acids. (see Fig. 11 for details).

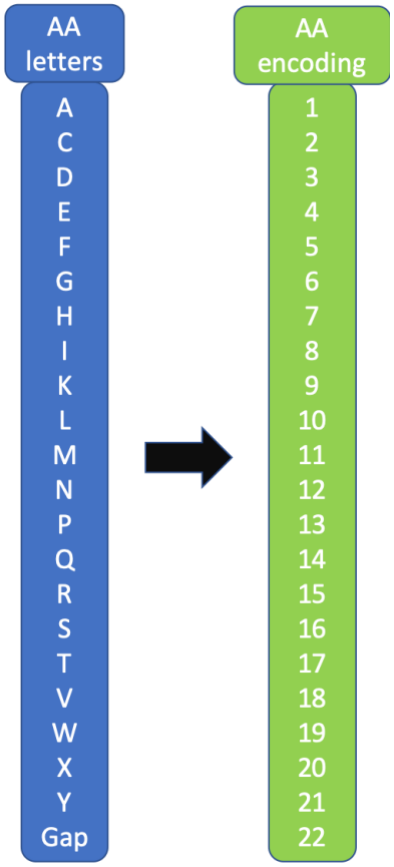



**Fig. 11 AA encoding**

## 3.2 Embedding Layer

The embedding phase mainly maps the input sequence into a matrix vector form, and each column corresponds to a word. That is, each number in the input sequence is mapped into a vector with a fixed length, and the input sequence is mapped into a matrix form of m × n. Among them, m is the embedding vector dimension, and n is the sequence length. The role of the embedding layer is to amplify some key features or separate some general features and map the digital sequence into a matrix vector form that is easy to process by the convolution layer, so that the subsequent convolution layer can fully extract the features. (See Fig. 12)

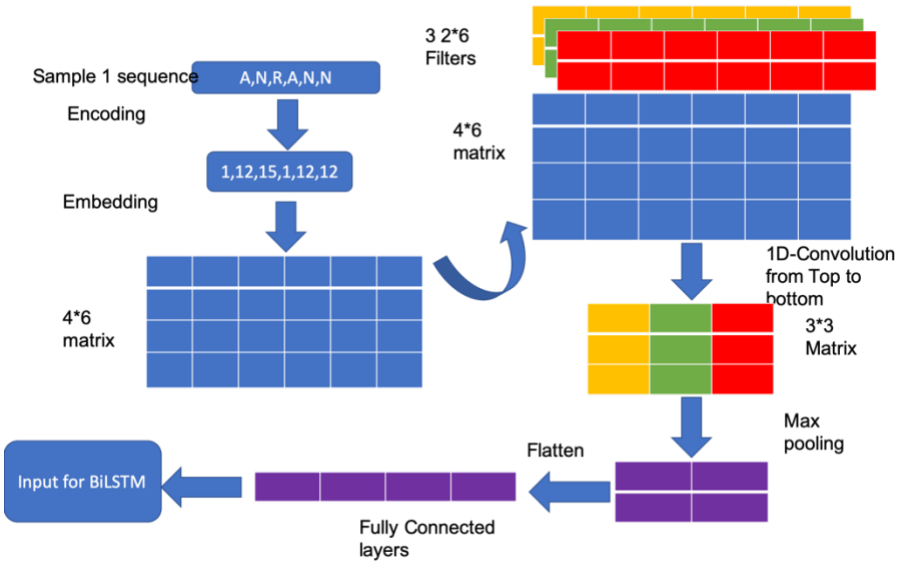



**Fig. 12**

**Embedding and convolution**

Note that the maximum length of protein sequences in our dataset is 91. To simplify this concept, we assume that the maximum sequence length is 6 and take the sequence Seq =ANRANN as an example.
First, we encoding the sequence to number form.
Next, the sequence is transformed into a multidimensional matrix by embedding.
In the convolutional layer, we use the 3 filters (2*6 Matrix) to scan 4*6 Matrix and obtain 3*3 matrix.
In the pooling layer, we adopt the max pooling method. This method adopts the maximum value of two numbers as their representative.
After that we extend our matrix into one-dimension.
Finally, add a fully connected layers after flatten layer and concatenate with non-AA position features as a new input layer for FCNN.

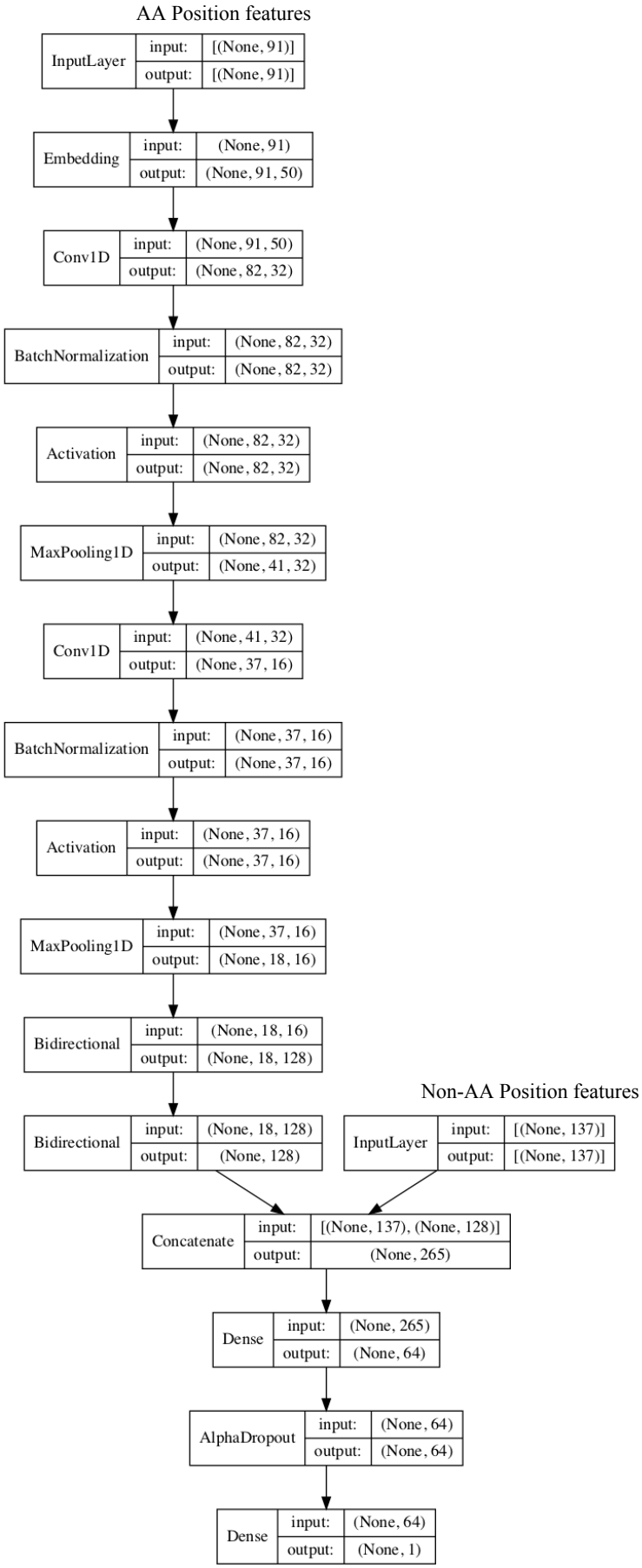



**Fig. 13 Flow plot for 1D-CNN + BiLSTM**

## 4. Hyperparameter tuning

We used grid search methods to do the hyperparameters tuning. For random forest model, we just set the tree number to be 250, max-depth is 25. We also set a random seed to make sure the results are reproducible.

## 5. Software information

We used python 3.7 and anaconda platform. We used Tensorflow 2.0 to construct our DNN models.

# Reference


1. https://www.who.int/hiv/data/en/.
2. WHO. WHO; 2019; Available from: https://www.who.int/news-room/fact-sheets/detail/hiv-aids.
3. Pebody R. How HIV works. 2019 [cited 2019]; Available from: http://www.aidsmap.com/about-hiv/hiv-1-and-hiv-2.
4. NIH. HIV Clinical Guidelines. NIH; 2019; Available from: https://aidsinfo.nih.gov/guidelines.
5. Spinner CD, Boesecke C, Zink A, Jessen H, Stellbrink HJ, Rockstroh JK, et al. HIV pre-exposure prophylaxis (PrEP): a review of current knowledge of oral systemic HIV PrEP in humans. Infection. 2016; 44:151-8.
6. Mayo Clinic Contributors. Mayo Clinic Minute: HIV Testing Day - Know your status. Mayo Clinic; 2019; Available from: https://www.mayoclinic.org/diseases-conditions/hiv-aids/diagnosis-treatment/drc-20373531.
7. CDC. HIV Vaccines. hiv.gov; 2017; Available from: https://www.hiv.gov/hiv-basics/hiv-prevention/potential-future-options/hiv-vaccines.
8. NIAID. History of HIV Vaccine Research. 2018; Available from: https://www.niaid.nih.gov/diseases-conditions/hiv-vaccine-research-history.
9. NIAID. Infographic: Progress Toward an HIV Vaccine. 2018; Available from: https://www.niaid.nih.gov/diseases-conditions/infographic-hiv-vaccine.
10. NIAID. HIV Vaccine Development. 2019 [cited 2019 May 15]; Available from: https://www.niaid.nih.gov/diseases-conditions/hiv-vaccine-development.
11. NIH. NIH Launches Large Clinical Trials of Antibody-Based HIV Prevention Studies on Three Continents Could Have Broad Implications for HIV Prevention Research. 2016; Available from: https://www.niaid.nih.gov/news-events/nih-launches-large-clinical-trials-antibody-based-hiv-prevention.
12. Zhou T, Georgiev I, Wu X, Yang ZY, Dai K, Finzi A, et al. Structural basis for broad and potent neutralization of HIV-1 by antibody VRC01. Science. 2010; 329:811-7.
13. Wu X, Yang ZY, Li Y, Hogerkorp CM, Schief WR, Seaman MS, et al. Rational design of envelope identifies broadly neutralizing human monoclonal antibodies to HIV-1. Science. 2010; 329:856-61.
14. Wibmer CK, Moore PL, Morris L. HIV broadly neutralizing antibody targets. Curr Opin HIV AIDS. 2015; 10:135-43.
15. Gnanakaran S, Daniels MG, Bhattacharya T, Lapedes AS, Sethi A, Li M, et al. Genetic signatures in the envelope glycoproteins of HIV-1 that associate with broadly neutralizing antibodies. PLoS Comput Biol. 2010; 6:e1000955.
16. Gilbert PB. Interpretability and robustness of sieve analysis models for assessing HIV strain variations in vaccine efficacy. Statistics in medicine. 2001; 20:263-79.
17. Edlefsen PT, Gilbert PB, Rolland M. Sieve analysis in HIV-1 vaccine efficacy trials. Curr Opin HIV AIDS. 2013; 8:432-6.
18. Gilbert P, Self S, Rao M, Naficy A, Clemens J. Sieve analysis: methods for assessing from vaccine trial data how vaccine efficacy varies with genotypic and phenotypic pathogen variation. Journal of clinical epidemiology. 2001; 54:68-85.

19. Edlefsen PT, Rolland M, Hertz T, Tovanabutra S, Gartland AJ, deCamp AC, et al. Comprehensive sieve analysis of breakthrough HIV-1 sequences in the RV144 vaccine efficacy trial. PLoS Comput Biol. 2015; 11:e1003973.
20. West A. CATNAP
Compile, Analyze and Tally NAb Panels. 2019.
21. Yoon H, Macke J, West Jr AP, Foley B, Bjorkman PJ, Korber B, et al. CATNAP: a tool to compile, analyze and tally neutralizing antibody panels. Nucleic acids research. 2015; 43:W213-W9.
22. Wikipedia contributors. IC50. Wikipedia: Wikipedia, The Free Encyclopedia.; 2020.
23. Magaret CA, Benkeser DC, Williamson BD, Borate BR, Carpp LN, Georgiev IS, et al. Prediction of VRC01 neutralization sensitivity by HIV-1 gp160 sequence features. PLoS Comput Biol. 2019; 15:e1006952.
24. Angermueller C, Parnamaa T, Parts L, Stegle O. Deep learning for computational biology. Mol Syst Biol. 2016; 12:878.
25. Alipanahi B, Delong A, Weirauch MT, Frey BJ. Predicting the sequence specificities of DNA- and RNA-binding proteins by deep learning. Nat Biotechnol. 2015; 33:831-8.
26. Todd CA, Greene KM, Yu X, Ozaki DA, Gao H, Huang Y, et al. Development and implementation of an international proficiency testing program for a neutralizing antibody assay for HIV-1 in TZM-bl cells. Journal of Immunological Methods. 2012; 375:57-67.
27. LeCun Y, Bottou L, Bengio Y, Haffner P. Gradient-based learning applied to document recognition. Proceedings of the IEEE. 1998; 86:2278-324.
28. Yih W-t, Toutanova K, Platt JC, Meek C. Learning discriminative projections for text similarity measures. Proceedings of the fifteenth conference on computational natural language learning; 2011: Association for Computational Linguistics.
29. Zeng H, Edwards MD, Liu G, Gifford DK. Convolutional neural network architectures for predicting DNA–protein binding. Bioinformatics. 2016; 32:i121-i7.
30. Zhang Q, Zhu L, Bao W, Huang D-s. Weakly-supervised convolutional neural network architecture for predicting protein-DNA binding. IEEE/ACM transactions on computational biology and bioinformatics. 2018.
31. Murugan P. Feed forward and backward run in deep convolution neural network. arXiv preprint arXiv:171103278. 2017.
32. Mikolov T, Karafiát M, Burget L, Černocký J, Khudanpur S. Recurrent neural network based language model. Eleventh annual conference of the international speech communication association; 2010.
33. Hochreiter S, Schmidhuber J. Long short-term memory. Neural computation. 1997; 9:1735-80.
34. Chawla NV, Bowyer KW, Hall LO, Kegelmeyer WP. SMOTE: synthetic minority over-sampling technique. Journal of artificial intelligence research. 2002; 16:321-57.
35. Lin T-Y, Goyal P, Girshick R, He K, Dollár P. Focal loss for dense object detection. Proceedings of the IEEE international conference on computer vision; 2017.

36. Li Y, Gal Y. Dropout inference in Bayesian neural networks with alpha-divergences. Proceedings of the 34th International Conference on Machine Learning-Volume 70; 2017: JMLR. org.
37. Kingma DP, Ba J. Adam: A method for stochastic optimization. arXiv preprint arXiv:14126980. 2014.
38. Ioffe S, Szegedy C. Batch normalization: Accelerating deep network training by reducing internal covariate shift. arXiv preprint arXiv:150203167. 2015.
39. Wikipedia contributors. Amino Acid. Wikipedia. Wikipedia, The Free Encyclopedia.2020.