San Jose State University
SJSU ScholarWorks

Master's Projects

Master's Theses and Graduate Research

Spring 5-22-2019

Predicting Off-Target Potential of CRISPR-Cas9 Single Guide RNA

Ishita Mathur
San Jose State University

Recommended Citation

Mathur, Ishita, "Predicting Off-Target Potential of CRISPR-Cas9 Single Guide RNA" (2019). *Master's Projects*. 718.
DOI: https://doi.org/10.31979/etd.r4d9-6y79
https://scholarworks.sjsu.edu/etd_projects/718

Predicting Off-Target Potential of CRISPR-Cas9 Single Guide RNA

A Project

Presented to

Dr. Sami Khuri

Department of Computer Science

San José State University

In Partial Fulfillment

Of the Requirements of the Degree

Master of Science

By

Ishita Mathur

May, 2019

ABSTRACT

Predicting Off-Target Potential of CRISPR-Cas9 Single Guide RNA

With advancements in the field of genome engineering, researchers have come up with potential ways for site-specific gene editing. One of the methods uses the Clustered Regularly Interspaced Short Palindromic Repeats - CRISPR-Cas technology. It consists of a Cas9 nuclease and a single guide RNA (sgRNA) that cleaves the DNA at the intended target site. However, the target genome could contain multiple potential off-target sites and cleaving an off-target site can have deleterious effects in case of gene editing in humans.

Lab based assays have been developed to test the off-target effects of guide RNAs. However, it is not feasible to scale these assays for reasons related to cost and labor. The use of Machine Learning models to compute the off-target potential makes these calculations cheaper and scalable. Both, classification as well as regression, can be used to solve this problem. In this project, we explore three classification models - Support Vector Machines (SVM), Logistic Regression and Convolutional Neural Networks (CNN).

ACKNOWLEDGEMENTS

I would like to express my gratitude to my advisor Dr. Sami Khuri for his support and guidance throughout the project. I would also like to thank my committee members Dr. Philip Heller and Dr. Wendy Lee for their support and time.

TABLE OF CONTENTS

CHAPTER 1 - INTRODUCTION

The discovery of the double helix structure of the DNA led to research in site-specific gene editing. Consequentially, the last half century has seen many advances in the field of Genome Engineering. Genome Engineering deals with modification, replacement, deletion or insertion of DNA in the genome of a living organism. The natural DNA repair mechanisms existing in organisms like yeast and bacteria showed that cells have ways to repair double stranded DNA breaks (DSB). The earlier methods to achieve DNA cleaving at a particular target used base pair recognition by oligonucleotides, a polynucleotide that contains a relatively small number of nucleotides [1]. These methods, while not robust, created the path for further experiments in targeted DNA cleaving.

1.1 Targeted Genome Editing

The first step in targeted genome editing is creating a DNA double stranded break (DSB). These DSBs can be repaired in two ways:

i. Non-homologous end joining (NHEJ)

ii. Homology directed repair (HDR)

In NHEJ, there is no homology template and the break ends are directly joined which can lead to indels of varying lengths causing frameshifts. HDR can be used by the cell only when the nucleus contains a homologue piece of DNA. It can be used to introduce point mutations or to insert specific sequences through the recombination of the target site with externally supplied DNA donor templates. Figure 1 illustrates the working of the two break repair methods.

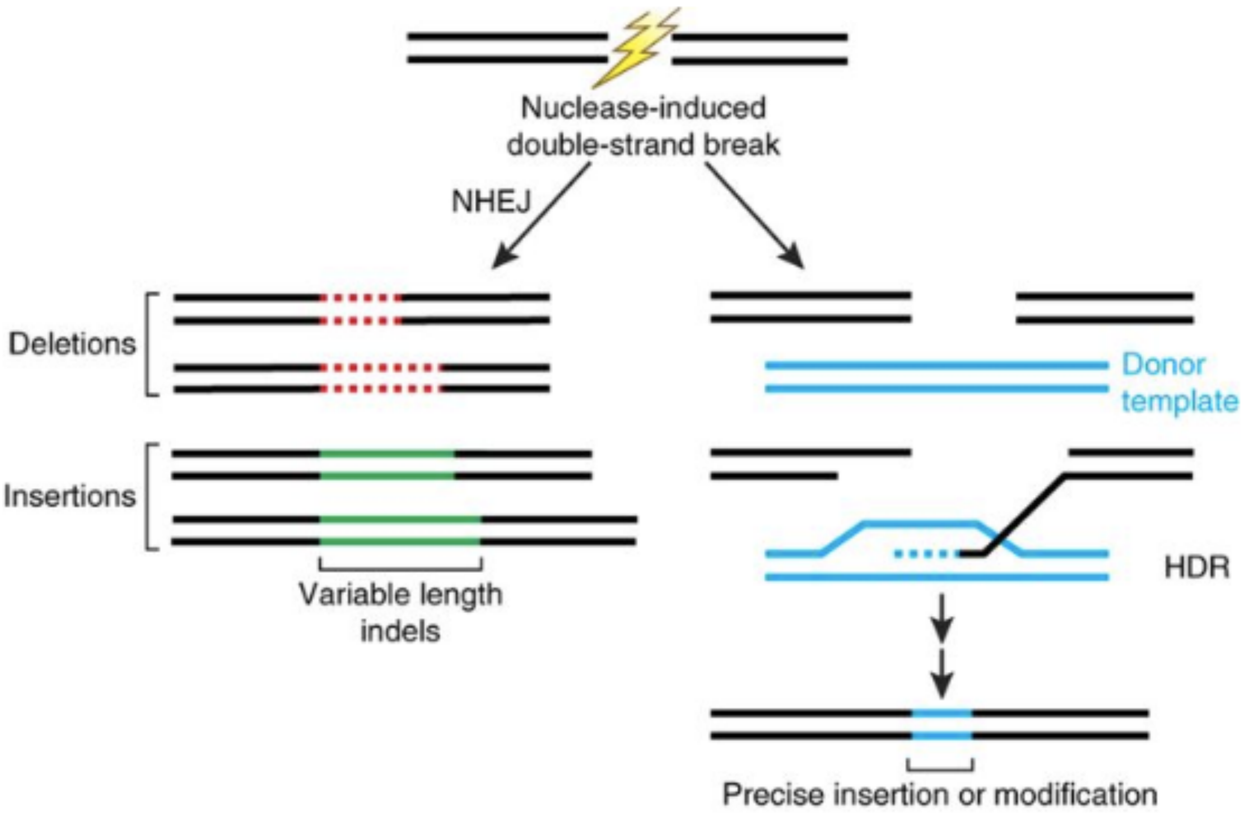

Nuclease-induced double-strand break
NHEJ
Deletions
Insertions
Variable length indels
Donor template
HDR
Precise insertion or modification

Figure 1. Double-strand Break Repair [2]

1.2 CRISPR-Cas System

In mid 2000s, researchers began to study Clustered Regularly Interspaced Short Palindromic Repeats (CRISPR), a naturally occurring DNA sequence found in some bacteria and archaea. The CRISPR array consists of repeaters interspaced with spacers. Cas (CRISPR-associated) is a collection of genes responsible for the functioning of CRISPR as a DNA cleaving tool. It was discovered that the CRISPR-Cas system may be responsible for adaptive immunity in bacteria which is achieved by cleaving the DNA of the attacking phage. The ability of the Cas9 system to bind to DNA at sites defined by the guide RNA and the PAM is an important attribute. This made researchers consider the possibility of using the CRISPR-Cas system in other fields of genetics such as agriculture and human gene therapy.

1.3 CRISPR-Cas System for Adaptive Immunity

The CRISPR-Cas system occurs naturally in some bacteria and archaea and provides adaptive immunity. CRISPR acts against invading genetic elements in the following three stages [2]:

A. Adaptation

Adaptation is the first stage of adaptive immunity. It involves three steps:

1. The bacterium is attacked by a foreign DNA
2. Cas genes bind to a part of the attacking DNA
3. A part of the attacking viral DNA is incorporated into the CRISPR array as a spacer flanked by the repeaters.

B. crRNA biogenesis

This stage is triggered by the host being attacked again. The CRISPR array is transcribed into a long precursor RNA in this step. The precursor is processed by a combination of Cas9, tracrRNA and RNaseIII to form a dsRNA, which is cleaved at one end by the RNaseIII. This releases the crRNAs from the precursor and produces Cas9 molecules that are ready to search invading DNA for targets.

C. Interference

The final stage involves the Cas:crRNA complex binding to the invading protospacer and cleaving the invading DNA.

Figure 2 shows the aforementioned steps involved in adaptive immunity [3]. Labels 'A', 'B' and 'C' demonstrate the first step - Adaptation; 'D', 'E', 'F' and 'G' demonstrate the second step - crRNA biogenesis; and 'H' and 'I' demonstrate the final step - Interference.

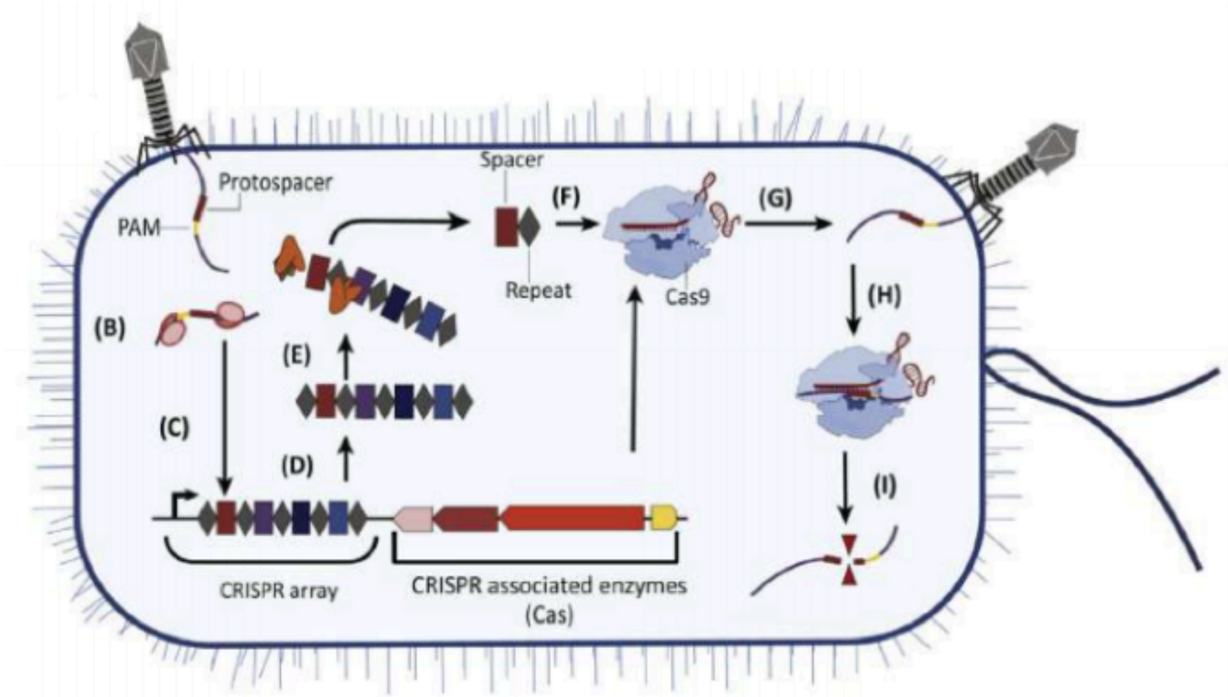


Figure 2. Adaptive Immunity [3]

The next chapter dives into more details of site specific editing.

CHAPTER 2 - SITE SPECIFIC EDITING

Four classes of customizable DNA have been developed so far to achieve site-specific gene editing [4]. These classes are:

i. meganucleases

ii. transcription activator-like effectors (TALEs)

iii. zinc finger (ZF) nucleases

iv. RNA guided Cas9 from type II CRISPR

Meganucleases, TALEs and ZF identify target sites through protein-DNA interactions. However, each has the following unique limitations:

i. Meganucleases aren't widely used due to lack of clear correspondence between meganuclease protein residue and their target DNA sequence specificity.

ii. ZF domains tend to have context-dependent binding preference due to crosstalk between adjacent modules when combined into an array

iii. TALE monomers can suffer from context-dependent specificity and construction of arrays is costly and labor intensive due to their repetitive sequences.

The Cas9 nuclease is guided by a guide RNA that is just 20-nt long. The guide RNA uses Watson-Crick base pairing to identify the target site.

2.1 CRISPR-Cas9 for site-specific gene editing

Editing using CRISPR-Cas requires only two components - Cas nuclease and a guide RNA. The guide sequence typically corresponds to the phage sequences but can be replaced by any sequence of interest. The target site has the protospacer adjacent motif (PAM) at the 3' end. The PAM sequence in *Streptococcus pyogenes* is 5'-NGG-3'.

2.2 Single guide RNA (sgRNA)

The Cas nuclease is guided to the target site by a single guide RNA. The guide RNA is 100-nt long but by altering 20 nucleotides towards its 5' end, it can be targeted towards complementary genomic sequence[5]. It is created from a tracrRNA:crRNA complex. The crRNA defines the target for Cas9 while the tracrRNA acts as a scaffold linking the crRNA to Cas9 and facilitates processing of mature crRNAs from pre-crRNAs. This creation of one complex makes experimental design simple. Figure 3 shows a schematic of Cas9 nuclease and the single guide RNA.

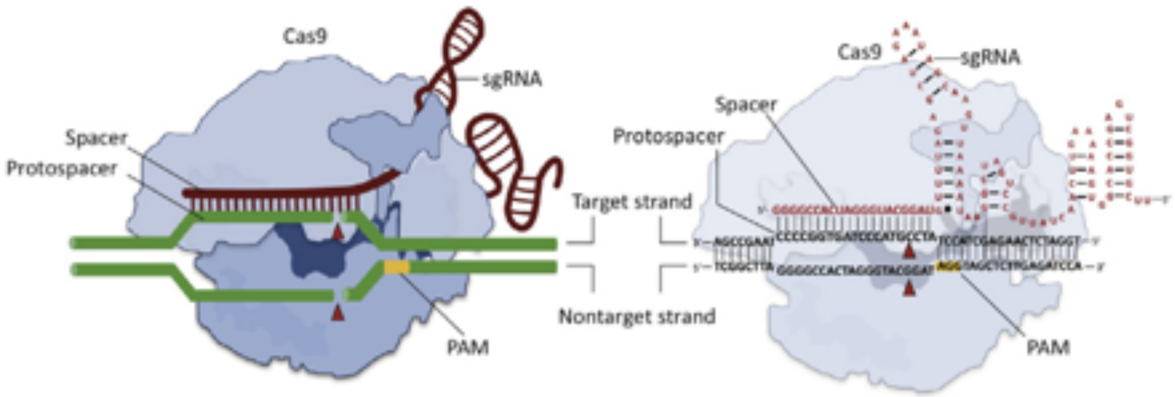


Figure 3. Cas9 nuclease and single guide RNA [3]

2.3 Off-target

An important factor when using CRISPR-Cas9 as a gene editing tool and when designing guide RNAs is the extent of off-target effects [6],[7]. The guide RNA sequence is designed to be complementary to the target site sequence. However, there can be multiple sites on the genome with a similar sequence (with some mismatches) where the guide RNA can bind. Some applications of genome engineering have a higher tolerance for off-target effects. However, applications like gene therapy in humans have a much lower tolerance for the same and off-target activities can have deleterious effects [7].

The next chapter explains the existing implementations of quantifying off-target effects to evaluate single guide RNAs.

CHAPTER 3 - EXISTING IMPLEMENTATIONS

The existing implementations to calculate off-target effects or potential of guide RNAs use lab based approaches as well computational methods using Machine Learning. This chapter details the various methodologies.

3.1 CHOP CHOP [8]

CHOP CHOP is a web tool that was developed to aid the designing of sgRNAs. It uses Bowtie sequence alignment to map candidate target sites into a sub region of the target site. The sgRNA pairs are then ranked based on the following factors:

i. number of off-targets in the genome

ii. number of mismatches within the off-target

iii. GC content

iv. presence of guanine at position 20 in the target site.

Target sites that receive the same score are further ranked according to their position in the gene where 5' gets higher preference.

3.2 Cutting Frequency Determination (CFD) [9]

Doench et. al. targeted a library containing the coding sequence of human and mouse CD33 with all single guide RNAs regardless of PAM. For all sites with the canonical NGG PAM, three types of mutations were added - 1 nucleotide deletions, 1 nucleotide insertions and 1 nucleotide mismatches. These mutations, in addition to complete match single guide RNAs, generated a library with 28,897 unique single guide RNAs.

CFD score is calculated by using percentage activity values. It is based on Naive Bayes, albeit with three assumptions. Given the equation: CFD ≡ $\prod_{i\in\{i|X_i=1\}} P(Y = 1|X_i = 1)$

where Y = 1 or 0 (active or inactive pair),

X_i = 1 or 0 (one hot encoding of features), the three assumptions are as follows:

i. conditioned on a guide RNA being active, features X_i are independent

ii. features are marginally independent

iii. $P(Y=1|X_i=0) = 1$.

3.3 DeepCRISPR[10]

Chuai et al. [10], created a hybrid neural network consisting of two parts - a pre trained deep convolutional de-noising neural network (DCDNN) based network and a convolutional neural network. The output from DCDNN was sent to the CNN. Each gRNA-target pair was encoded in two parts and one pair from each gRNA was treated as the sample locus. Each part of the sample was fitted into the pre trained DCDNN model and the outputs were combined channel wise for the CNN classifier. The training procedure learned weights for the classifier as well as fine-tuned weights for the parent network. An anti OT score was calculated at the end for each gRNA as:

$$S = \frac{\ln(1 + e^{\Sigma(OT_i)})}{\ln 2}$$

where OT_i is the occurrence probability of target site i.

The gRNAs are ranked in descending order of score.

3.4 Elevation[7]

Listgarten et al. developed the tool "Elevation" as an improvement over CFD. The tool uses Machine Learning to evaluate guide RNAs and consists of three basic steps:

1. Elevation Search - search and filter genome wide for potential target sites for one single guide RNA. It uses two tandem seeds and an extension to find potential target sites which are then organized into a tree data structure.

2. Elevation Score - score each potential target for activity. It is a two layer model. The first layer considers single mismatches in each sgRNA-target pair and the second layer combines the results from the first layer. The first layer uses Gradient Boosted Regression Trees and the second layer uses L1-regularized Linear Regression. The results are passed through a Logistic Regression model for calibration.

3. Elevation Aggregate - aggregate scores from step 2 into a single off-target prediction value to assess the single guide RNA. It uses the results from Elevation Score and combines them to produce a final score for each sgRNA. This layer uses Gradient Boosted Regression Trees to compute the aggregate score.

Elevation uses the GUIDE-Seq [11] dataset for training and validation.

The next chapter introduces the machine learning and deep learning techniques used in this project.

CHAPTER 4 - MACHINE LEARNING MODELS

Machine Learning is the study of algorithms and statistical models that computer systems use to improve their performance. Machine learning algorithms create a mathematical model using a set of data called “training data” to make predictions or decisions on a different set of data called “test data” without being explicitly programmed to do so. Deep Learning is a part of Machine Learning based on learning data representations. There are three types of learning algorithms - supervised, unsupervised and reinforcement. This project utilizes and compares two Machine Learning models - Support Vector Machines and Logistic Regression, and one Deep Learning model - Convolutional Neural Networks. All three models classify as supervised learning.

4.1 Support Vector Machines (SVM)

Support Vector Machines are supervised learning models that are used for classification and regression[12]. This project uses SVM as a classifier. The SVM classifier aims to construct a hyperplane or a set of hyperplanes that separates the data points into the two classes (in case of binary classification, as in this project), while maximizing the margin between both classes. Figure 4 shows a hyperplane separating two classes while maximizing the minimum distance from each class.

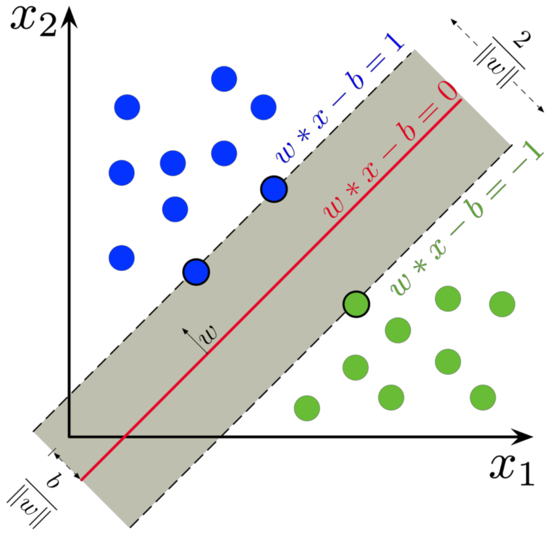


Figure 4. SVM Classification with Hyperplane[12]

4.2 Logistic Regression

Logistic Regression is a form of binomial regression [13]. It involves estimating the parameters of a logistic model. It has been used in biological sciences as well as social sciences. Logistic Regression can be used when the output is categorical, hence also functioning as a binary classifier[14]. It uses the sigmoid function to compute the probability of the data belonging to either class. The sigmoid function is given as: S(x) = $\frac{1}{1+ e^{-x}}$. Figure 5 shows the graph of a sigmoid function.

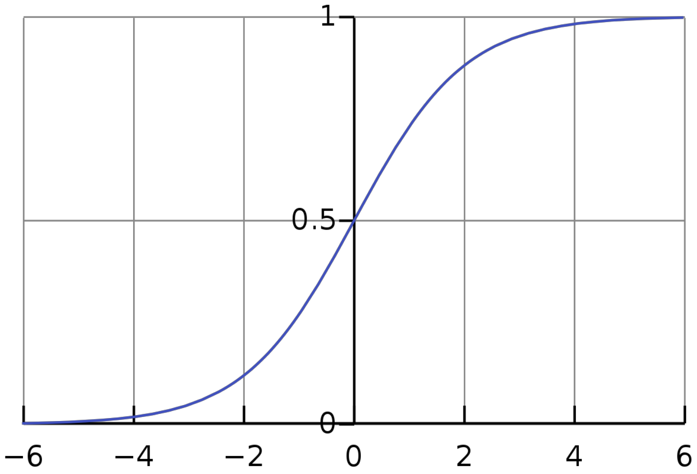


Figure 5. Graph of a Sigmoid Function [15]

4.3 Convolutional Neural Networks (CNN)

A Convolutional Neural Network is a Deep Learning algorithm that can be used for classification [16]. CNNs classify as a supervised learning algorithm. They can take an image, in the form of a matrix, as an input and assign importance to different features. CNNs are modeled after the neuron connectivity in human brain. They consist of multiple layers connected to each other. The following subsections describe the different types of layers and functions that are commonly used in CNNs.

4.3.1 Convolutional Layer

This layer consists of one or more filter matrices which can be applied to the input matrix to obtain a convoluted matrix. Different filters are available in the model to cater to different features.

4.3.2 Activation Function

Activation function decides the output of the node for the given input. There are multiple activation functions that cater to different types of problems.

4.3.3 Batch Normalization Layer

Batch Normalization layer normalizes the inputs of each layer such that mean activation value is 0 and standard deviation is 1.

4.3.4 Max Pooling Layer

It partitions the input matrix into a set of non overlapping rectangles and outputs the maximum for each region. It helps reduce the spatial size of the image representation.

4.3.5 Dense Layers

Dense layers are fully connected layers where the neurons in each layer are connected to all activations in the previous layer.

4.3.6 Dropout Layer

The dropout layer is a regularization layer, connected to the dense layers. It is used to avoid overfitting.

4.3.7 Softmax Function

This function calculates the probability of each target class over all possible target classes. These probabilities are later used to find the target class for the image. Softmax function can be given mathematically as: $\sigma(z)_j = \frac{e^{(z_j)}}{\sum_{k=1}^{K} e^{z_k}}$ for j = 1,2,...,K

The next chapter explains the data collection and feature extraction process.

CHAPTER 5 - DATA COLLECTION AND FEATURES

5.1 Data Collection

Tsai et al. [11] developed the GUIDE-Seq database using lab based experiments combined with Burrows Wheeler Alignment. They filtered out off-target cleavage sites with more than six mismatches. The GUIDE-Seq pipeline was performed on U2OS and HEK293 cells and the resulting dataset contains the identified off-target sites.

Listgarten et al. [7] used the GUIDE-Seq dataset to build their tool "Elevation" and further added sequences from human and mouse CD33. They also introduced mutations to the CD33 dataset in three ways - 1 nucleotide insertions, 1 nucleotide deletions and 1 nucleotide mismatches generating an extensive dataset with CD33 guides and identified off-targets.

This project uses the Elevation dataset, consisting of the CD33 sequences to train the models. This dataset contains 40,270 records with both human and mouse targets with 78 unique human targets and 20 unique mouse targets. The models are tested on the CRISPOR [17] dataset.

Since all datasets contain only identified off-target sites, the training data was imbalanced. To create a balanced dataset, the non off-target sites were found by scanning the genomes. For each target site, the target genome was scanned for 20 nucleotide long sites that had more than 6 mismatches to the intended target site. These sequences were then collected and added to the dataset to balance it.

5.2 Features

There are mainly two types of features that are extracted from the sgRNA-target site pair:

i. positions of mismatch

ii. identity of mismatch

Identity of a mismatch can take two values - transition or transversion, represented as 1 and 2 respectively, with 0 representing no mismatch.

For the Support Vector Machine and the Logistic Regression models, each sgRNA-target pair is one hot encoded as a vector of length 40. The first 20 features correspond to each position in the sequence and a ‘1’ indicates a mismatch. The last 20 features correspond to each position in the sequence and ‘1’ indicates a transition and ‘2’ indicates a transversion at the site of a mismatch.

For Convolutional Neural Networks, the sgRNA-target pair is encoded as a matrix and logical OR is performed between both matrices to obtain a single matrix containing the mismatch as well as identity of the mismatch.

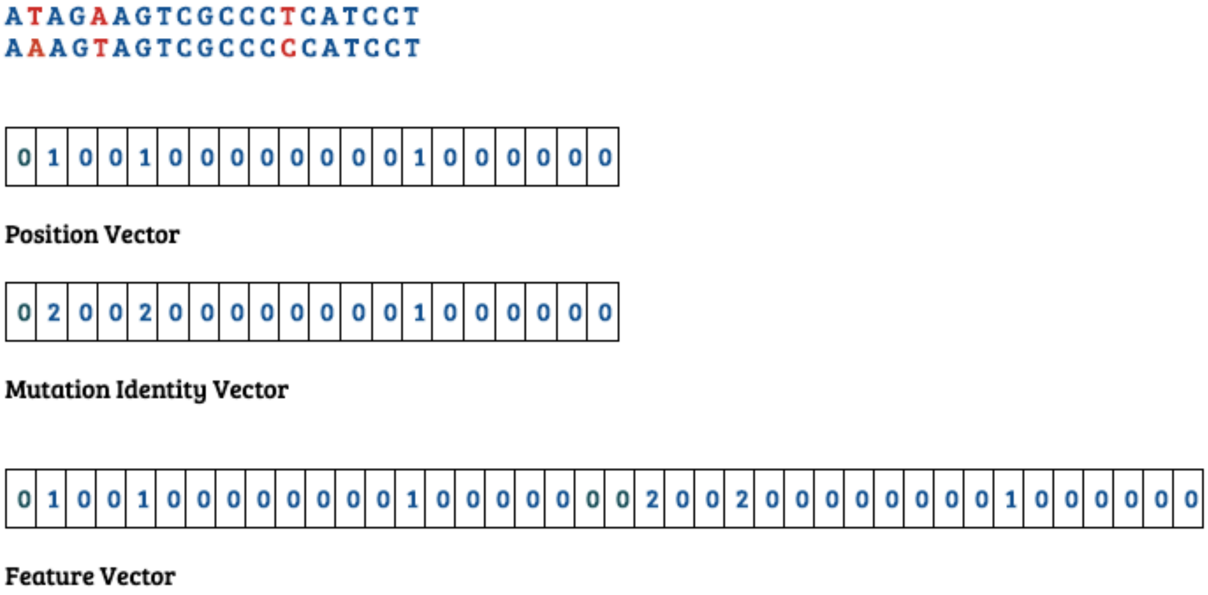
The next chapter details the methods used in this project.

CHAPTER 6 - METHODS

6.1 Support Vector Machines (SVM)

6.1.1 Training Data

The CD33 data from the Elevation [7] paper was used as training data. Feature extraction, as described in the previous chapter, was performed on each sgRNA-target pair. For each pair, a feature vector of length 40 was obtained. Figure 6 shows a sample feature vector.



ATAGAAGTCGCCCTCATCCT
AAAGTAGTCGCCCCCATCCT

0	1	0	0	1	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0

Position Vector

0	2	0	0	2	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0

Mutation Identity Vector

0	1	0	0	1	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	2	0	0	2	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0

Feature Vector

Figure 6. Sample Feature Vector

6.1.2 SVM Classifier

This project uses the SVC classifier in the scikit-learn package to classify the data. SVC is used with the default parameters. Additionally, “probability” is set to “True” and “kernel” is changed to “linear” to allow Recursive Feature Elimination (RFE).

6.1.3 Training the Classifier

The training data was split into training and validation data. Model accuracy was improved using Recursive Feature Elimination (RFE). RFE involved finding coefficients for each feature. In each iteration, the feature with the lowest coefficient is eliminated till model accuracy stops increasing. Figure 7 shows the process of training the classifier.

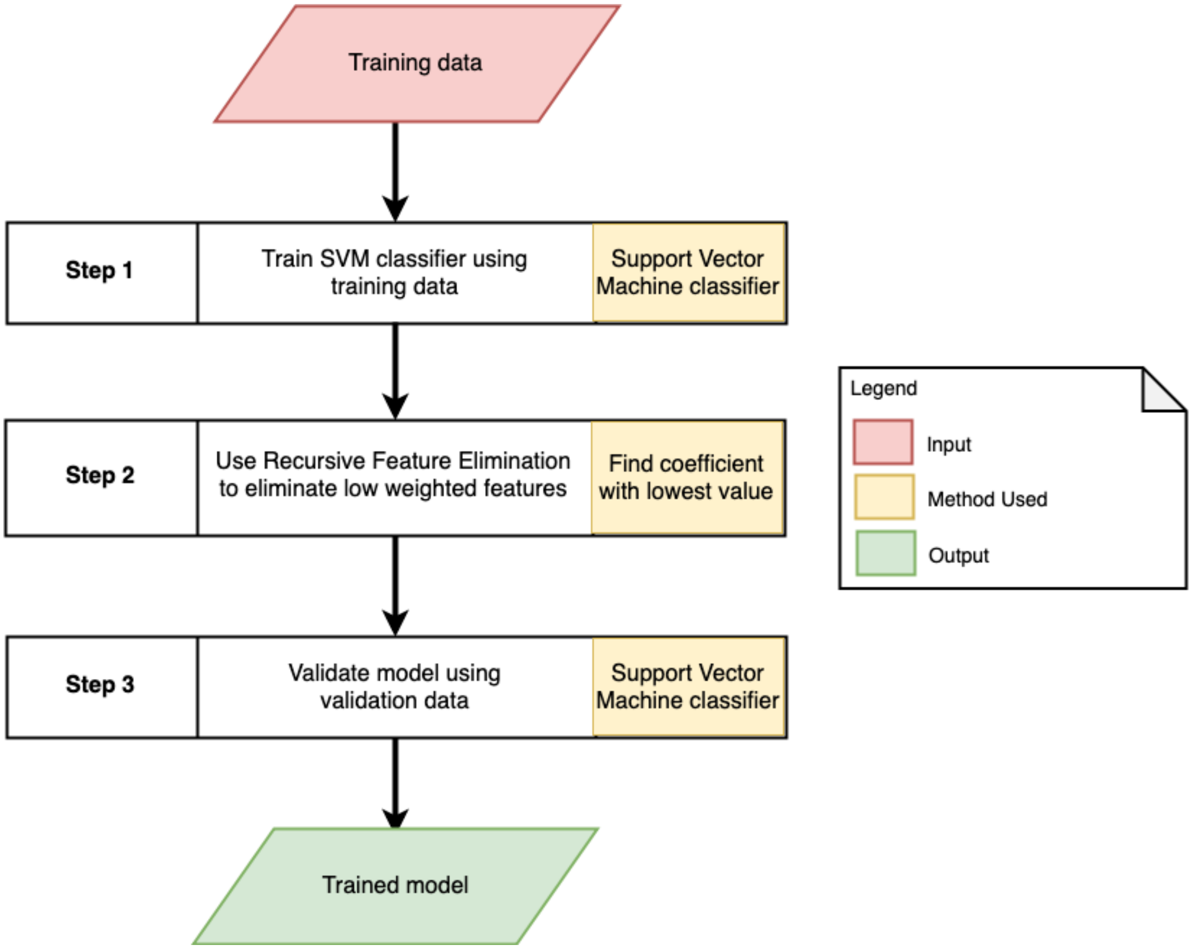

Training data
Step 1
Train SVM classifier using training data
Support Vector Machine classifier
Step 2
Use Recursive Feature Elimination to eliminate low weighted features
Find coefficient with lowest value
Step 3
Validate model using validation data
Support Vector Machine classifier
Trained model
Legend
Input
Method Used
Output

Figure 7. SVM Classifier Training

6.1.4 Aggregate Score Calculation

For each test sgRNA, 20 nucleotide sites are found on the target genome. Feature extraction is performed on each sgRNA-target pair and passed to the trained model. The model classifies each site as either off-target (1) or not off-target (0). The probabilities of all sites classified as off-target are then used to calculate the aggregate score for the sgRNA.

The aggregate score for each sgRNA is calculated using the formula used by Chuai et al. for DeepCRISPR[10]:

$S = \ln (1 + e^{\sum p_i}) / \ln 2$

where p_i is the probability of each target site classified as an off-target, being an off-target site.

Figure 8 shows the process involved in aggregate score calculation using SVM classifier.

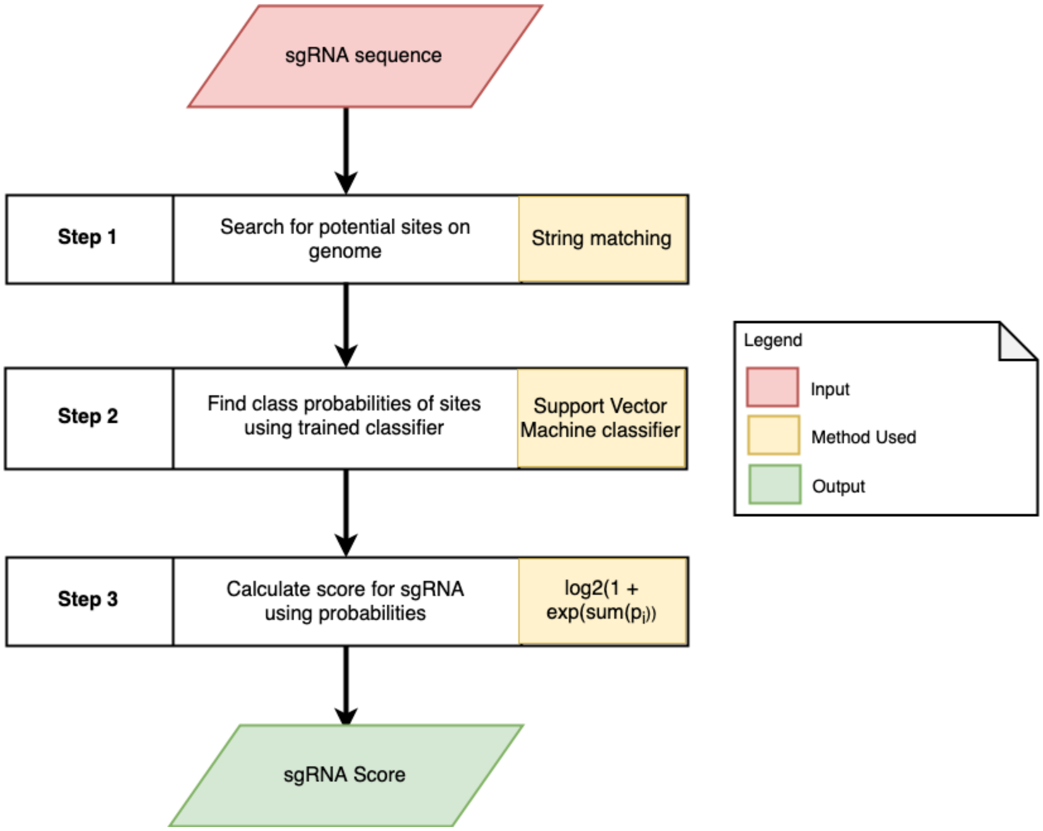

sgRNA sequence
Step 1
Search for potential sites on genome
String matching
Step 2
Find class probabilities of sites using trained classifier
Support Vector Machine classifier
Step 3
Calculate score for sgRNA using probabilities
log2(1 + exp(sum(p_j))
sgRNA Score
Legend
Input
Method Used
Output

Figure 8. Using Support Vector Machines to Calculate Off-target Score

6.2 Logistic Regression

6.2.1 Training Data

The Logistic Regression model is also trained on the CD33 data that was generated by the Elevation [11] paper. Feature extraction was performed as described in the previous chapter and illustrated in Figure 6.

6.2.2 Logistic Regression Model

This project uses Logistic Regression available in the scikit-learn package to build the model. The model is built with the default scikit-learn parameters.

6.2.3 Aggregate Score Calculation

For each test sgRNA, 20 nucleotide sites are found on the target genome. Feature extraction is performed on each sgRNA-target pair and passed to the trained model. The model classifies each site as either off-target (1) or not off-target (0). The probabilities of all sites classified as off-target are then used to calculate the aggregate score for the sgRNA.

The aggregate score for each sgRNA is calculated using the formula used by Chuai et al. for DeepCRISPR[10]:

$S = \ln (1 + e^{\sum p_i}) / \ln 2$

where p_i is the probability of each target site classified as an off-target, being an off-target site.

Figure 9 illustrates the process involved in aggregate score calculation using Logistic Regression for binary classification.

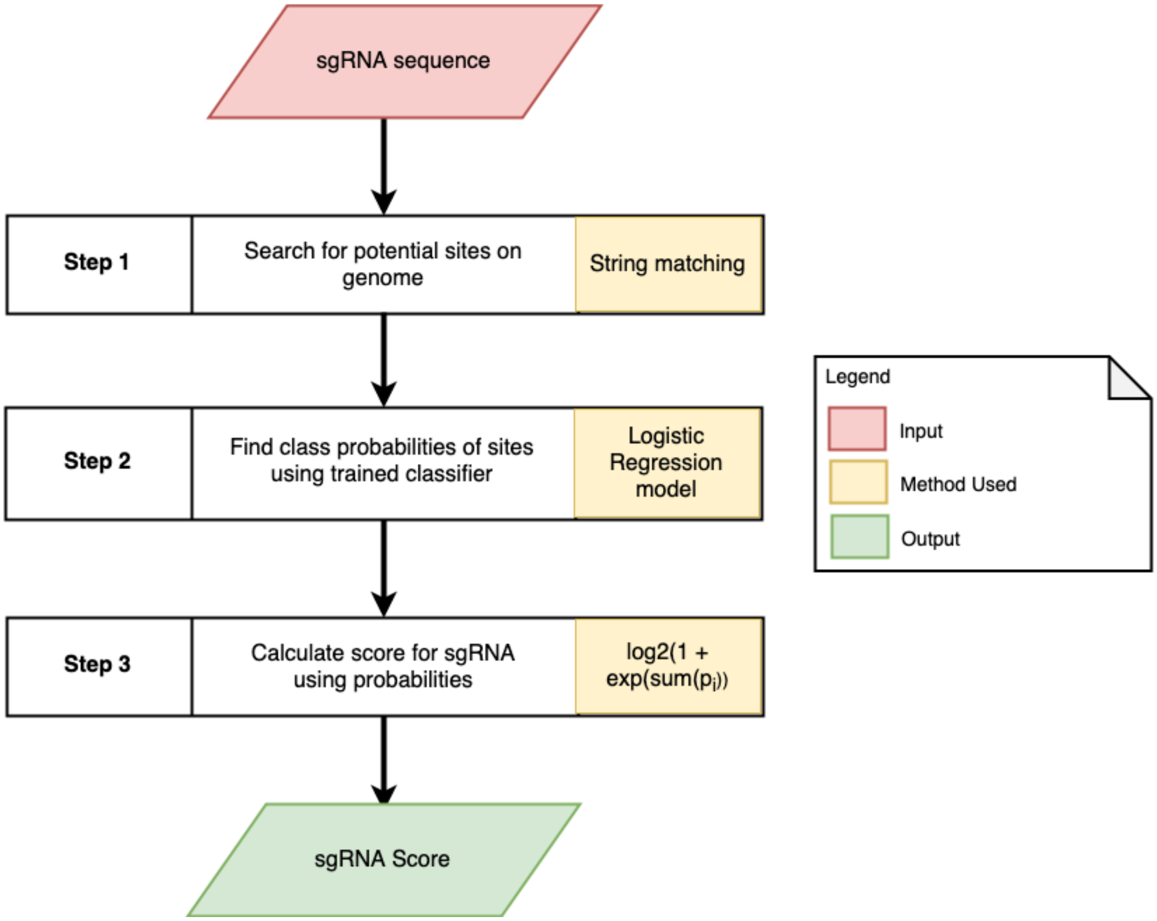
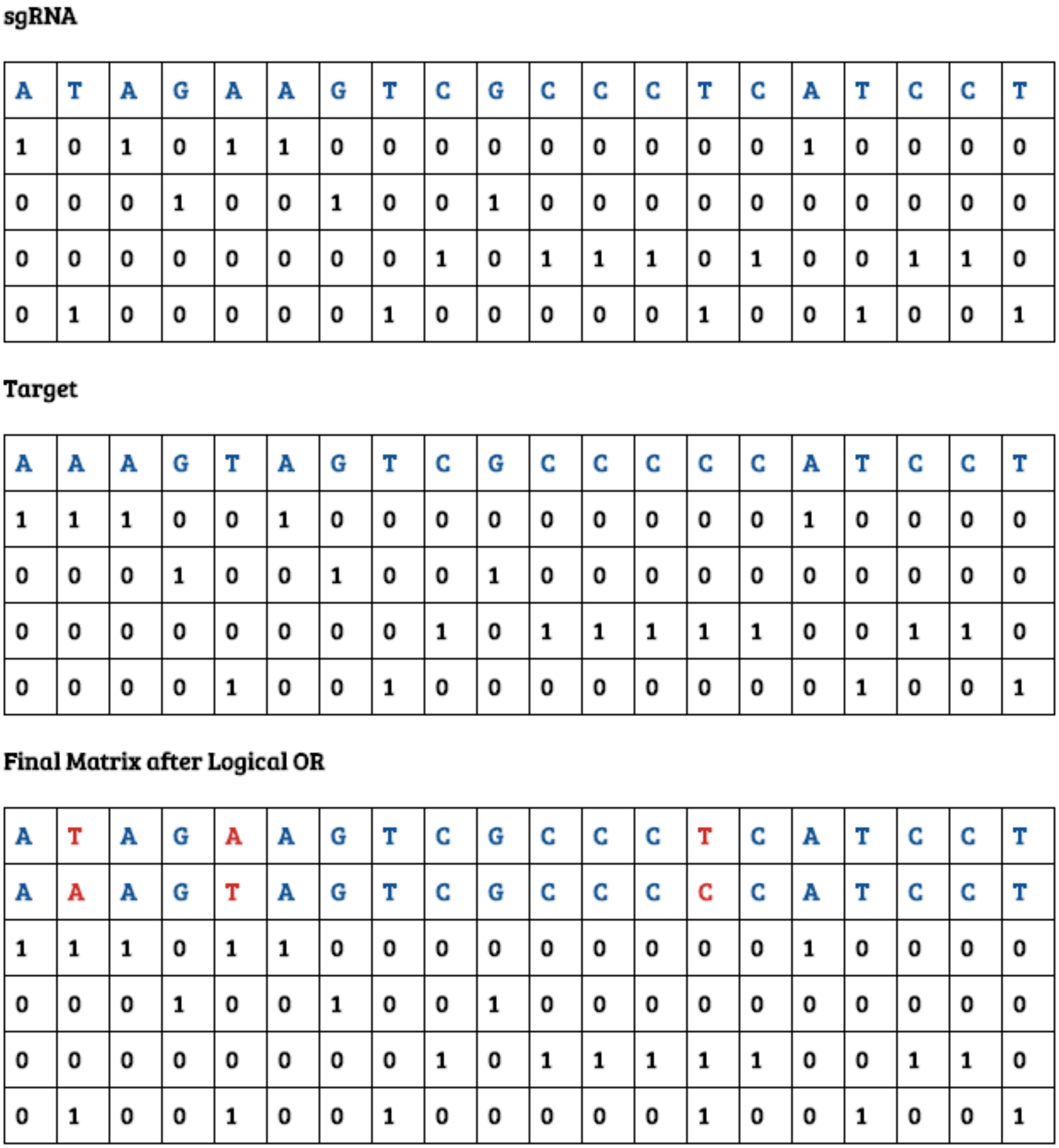

sgRNA sequence
Step 1
Search for potential sites on genome
String matching
Step 2
Find class probabilities of sites using trained classifier
Logistic Regression model
Step 3
Calculate score for sgRNA using probabilities
log2(1 + exp(sum(p_j))
sgRNA Score
Legend
Input
Method Used
Output

Figure 9. Logistic Regression to Calculate Off-target Score

6.3 Convolutional Neural Networks (CNN)

6.3.1 Training Data

The CD33 data from the Elevation [7] paper is used as training data. The input data is transformed into a 20x4 matrix before being passed to the model for training. Each sgRNA-target pair is one hot encoded individually into a 20x4 matrix. Logical OR is performed on both matrices to get a final input matrix for each pair, encoding the position as well as identity of the mismatch. Figure 10 illustrates the input matrix generation for a sample sgRNA-target pair.



sgRNA

A	T	A	G	A	A	G	T	C	G	C	C	C	T	C	A	T	C	C	T
1	0	1	0	1	1	0	0	0	0	0	0	0	0	0	1	0	0	0	0
0	0	0	1	0	0	1	0	0	1	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	1	0	1	1	1	0	1	0	0	1	1	0
0	1	0	0	0	0	0	1	0	0	0	0	0	1	0	0	1	0	0	1

Target

A	A	A	G	T	A	G	T	C	G	C	C	C	C	C	A	T	C	C	T
1	1	1	0	0	1	0	0	0	0	0	0	0	0	0	1	0	0	0	0
0	0	0	1	0	0	1	0	0	1	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	1	0	1	1	1	1	1	0	0	1	1	0
0	0	0	0	1	0	0	1	0	0	0	0	0	0	0	0	1	0	0	1

Final Matrix after Logical OR

A	T	A	G	A	A	G	T	C	G	C	C	C	T	C	A	T	C	C	T
A	A	A	G	T	A	G	T	C	G	C	C	C	C	C	A	T	C	C	T
1	1	1	0	1	1	0	0	0	0	0	0	0	0	0	1	0	0	0	0
0	0	0	1	0	0	1	0	0	1	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	1	0	1	1	1	1	1	0	0	1	1	0
0	1	0	0	1	0	0	1	0	0	0	0	0	1	0	0	1	0	0	1

Figure 10. Same Input Matrix for CNN

6.3.2 Model Architecture

Figure 11 depicts the architecture of the CNN model used for this project.

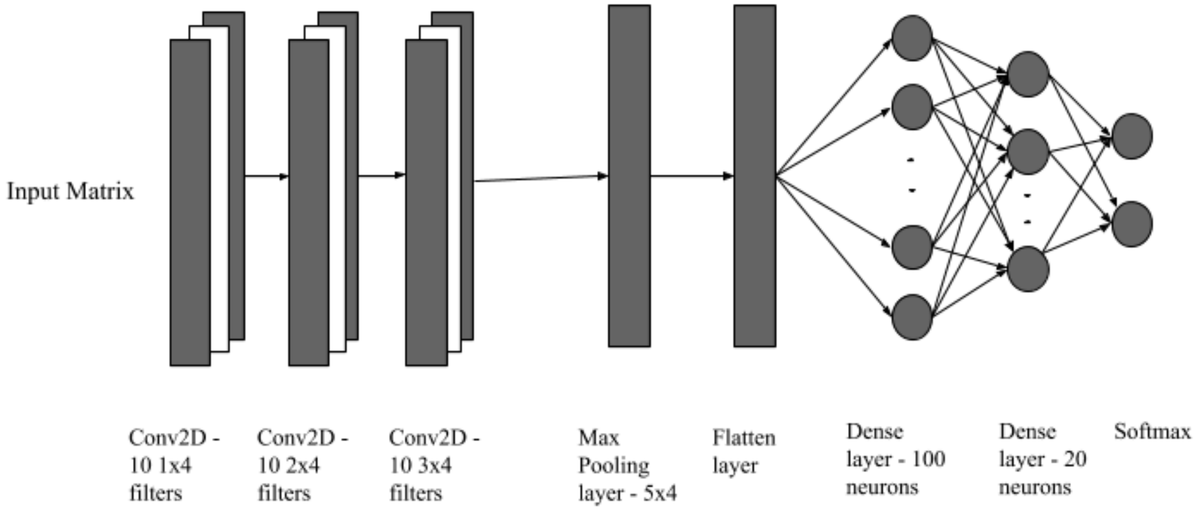


Figure 11. CNN Architecture

The different layers of the model are built using Keras [19]. There are three convolutional layers with the following properties:

i. The first convolutional layer contains 10 filters with kernel size 1x4.

ii. The second convolutional layer contains 10 filters with kernel size 2x4.

iii. The third convolutional layer contains 10 filters with kernel size 3x4.

Each filter has one of its dimensions set as 4 to maintain base-pair integrity of the input sgRNA-target pair. The three convolutional layers use the 30 filters to extract features from the input matrix. Each convolutional layer uses 'ReLU' as the activation function. ReLU stands for rectified linear unit and is given by the equation *y = max(0,x)*. Figure 12 shows the graph for the ReLU funtion.

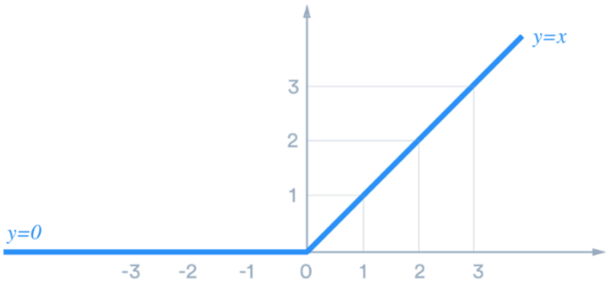
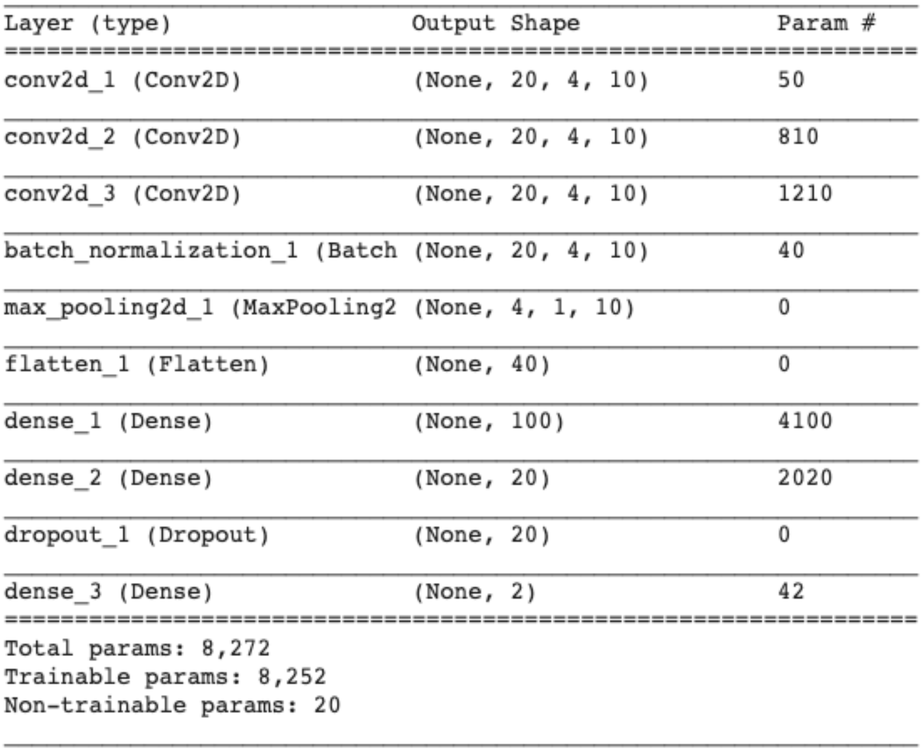


Figure 12. ReLU Function

The convolutional layers are followed by a max pooling layer. The max pooling layer contains a filter of size 5x4 and helps in dimensionality reduction. The resultant matrix from the max pooling layer is passed to a flatten layer which turns the matrix into a 1D vector. The 1D vector is passed through two dense layers with 100 and 20 neurons each. The last layer contains 2 neurons that use the Softmax function to give the probability of the site being an off-target or not an off-target. Figure 13 shows the output shape and number of parameters after each layer.



```
_________________________________________________________________
Layer (type)                 Output Shape              Param #   
=================================================================
conv2d_1 (Conv2D)            (None, 20, 4, 10)         50        
_________________________________________________________________
conv2d_2 (Conv2D)            (None, 20, 4, 10)         810       
_________________________________________________________________
conv2d_3 (Conv2D)            (None, 20, 4, 10)         1210      
_________________________________________________________________
batch_normalization_1 (Batch (None, 20, 4, 10)         40        
_________________________________________________________________
max_pooling2d_1 (MaxPooling2 (None, 4, 1, 10)          0         
_________________________________________________________________
flatten_1 (Flatten)          (None, 40)                0         
_________________________________________________________________
dense_1 (Dense)              (None, 100)               4100      
_________________________________________________________________
dense_2 (Dense)              (None, 20)                2020      
_________________________________________________________________
dropout_1 (Dropout)          (None, 20)                0         
_________________________________________________________________
dense_3 (Dense)              (None, 2)                 42        
=================================================================
Total params: 8,272
Trainable params: 8,252
Non-trainable params: 20
_________________________________________________________________
```

Figure 13. CNN Model Summary

6.3.3 Training the CNN

The CNN model was trained and validated using the CD33 data [7]. The mean squared error was used as the loss function and the Adam algorithm was used for optimization. The Adam algorithm maintains a learning rate for each network weight, which get separately adapted as the learning unfolds.

6.3.4 Aggregate Score Calculation

For each test sgRNA, 20 nucleotide sites are found on the target genome. Each sgRNA-target pair is encoded into a matrix, converted to the final matrix by performing a logical OR and passed to the trained model. For each pair the model returns the probability of the site being an off-target (1) and not an off-target (0). For each sgRNA, the probability of sites that return a higher probability of being an off-target are used to calculate the final score for the sgRNA.

The aggregate score for each sgRNA is calculated using the formula used by Chuai et al. for DeepCRISPR[10]:

$S = \ln(1 + e^{\sum p_i}) / \ln 2$

where p_i is the probability of each target site classified as an off-target, being an off-target site.

The next chapter goes over the results of each of these methods.

CHAPTER 7 - RESULTS

Each model was trained on the Elevation dataset and tested on the CRISPOR dataset. The Elevation dataset contains 6819 sgRNA-target pairs with 78 unique single guide RNAs from human and mouse CD33 genome, and the CRISPOR dataset consists of 720 sgRNA-target pairs with 31 unique single guide RNAs. Figure 14 shows the performance comparison of the three models.

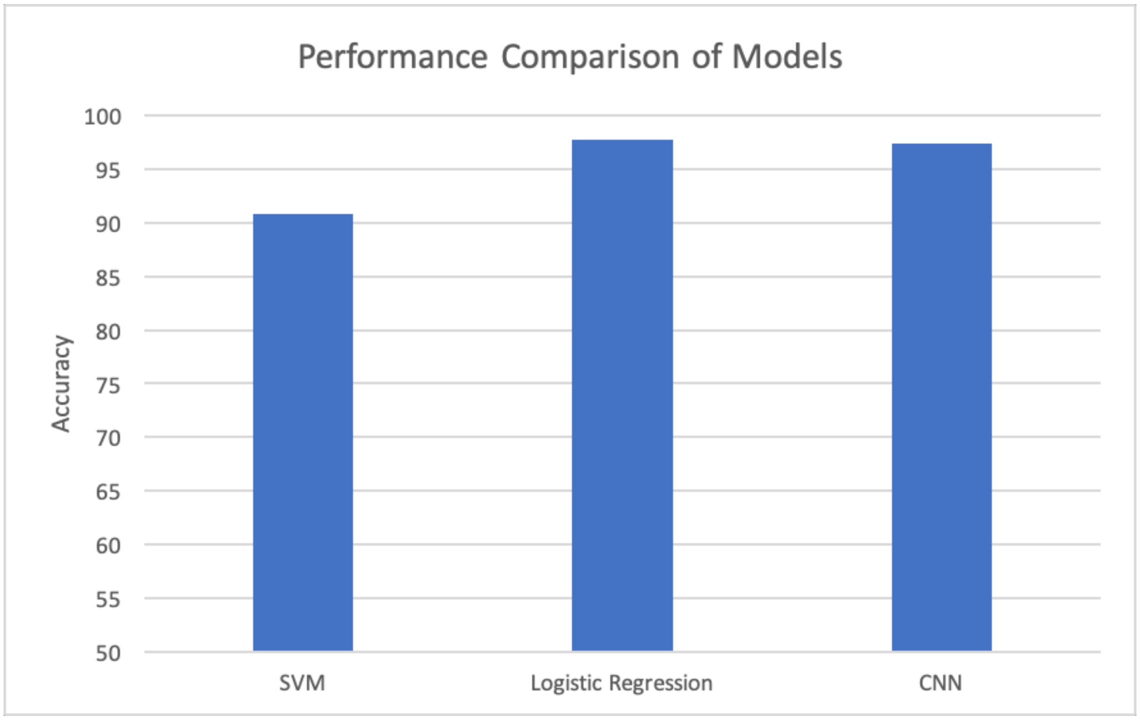


Figure 14. Performance Comparison of all Models

The Support Vector Machine classifier gives an accuracy of 90.2% and ROC-AUC of 0.85. Figure 15 shows the ROC curve and the AUC for the SVM classifier.

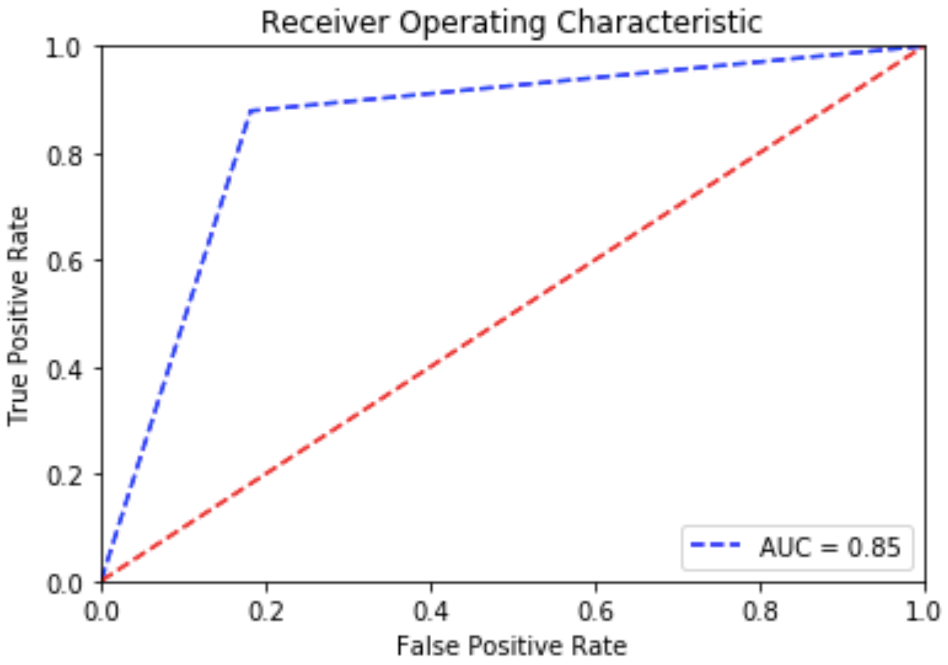


Figure 15. ROC Curve for SVM Classifier

Logistic Regression gives an accuracy of 97.77% and ROC-AUC of 0.90. Figure 16 shows the ROC curve for the Logistic Regression model.

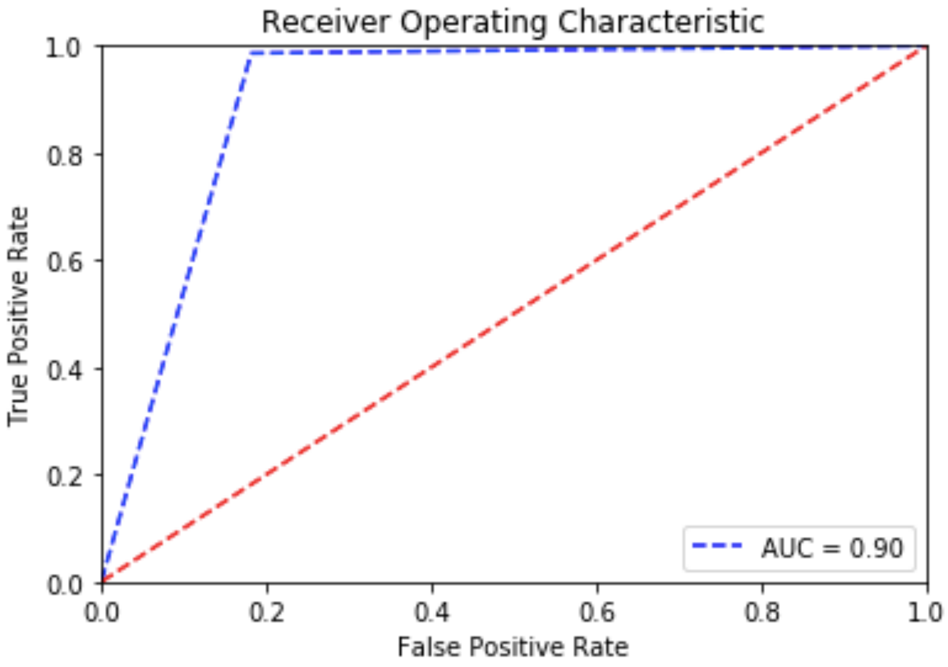


Figure 16. ROC Curve for Logistic Regression

The Convolutional Neural Network model gives an accuracy of 97.32% and ROC-AUC of 0.90 over 10 epochs. Figure 17 shows the ROC curve for the CNN model.

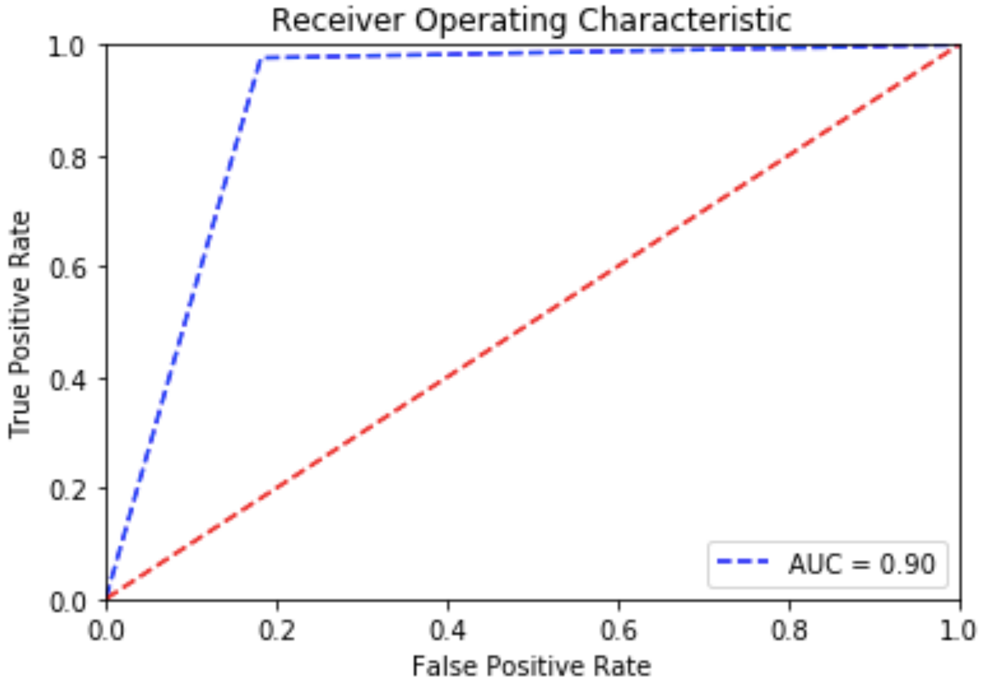


Figure 17. ROC Curve for Convolutional Neural Networks

Figures 18-21 illustrate the training loss, training accuracy, validation loss and validation accuracy against the number of epochs for the CNN model.

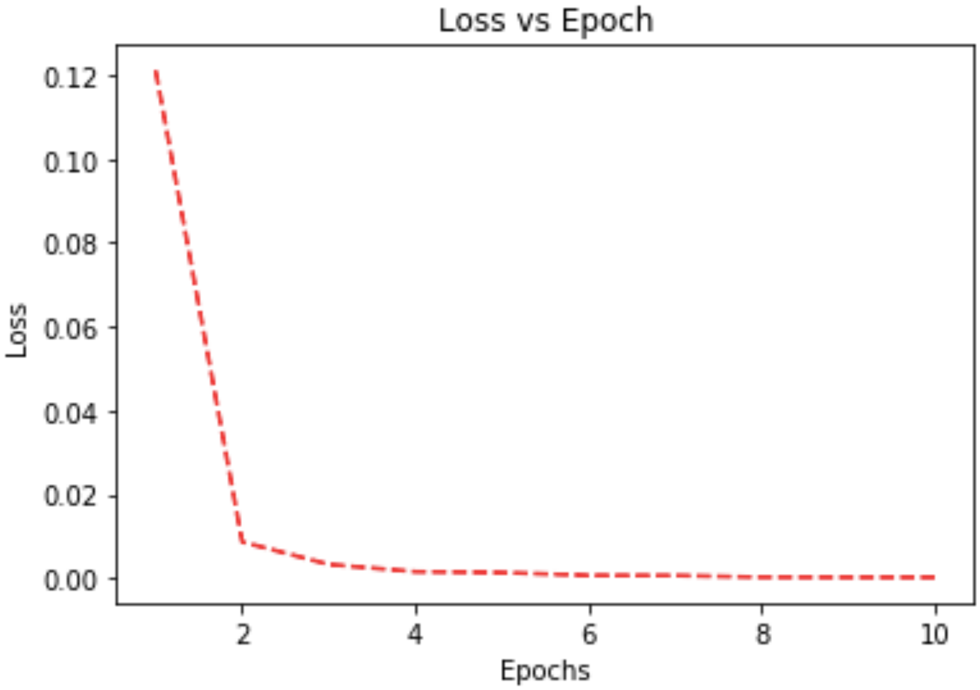


Figure 18. Training Loss vs Epochs

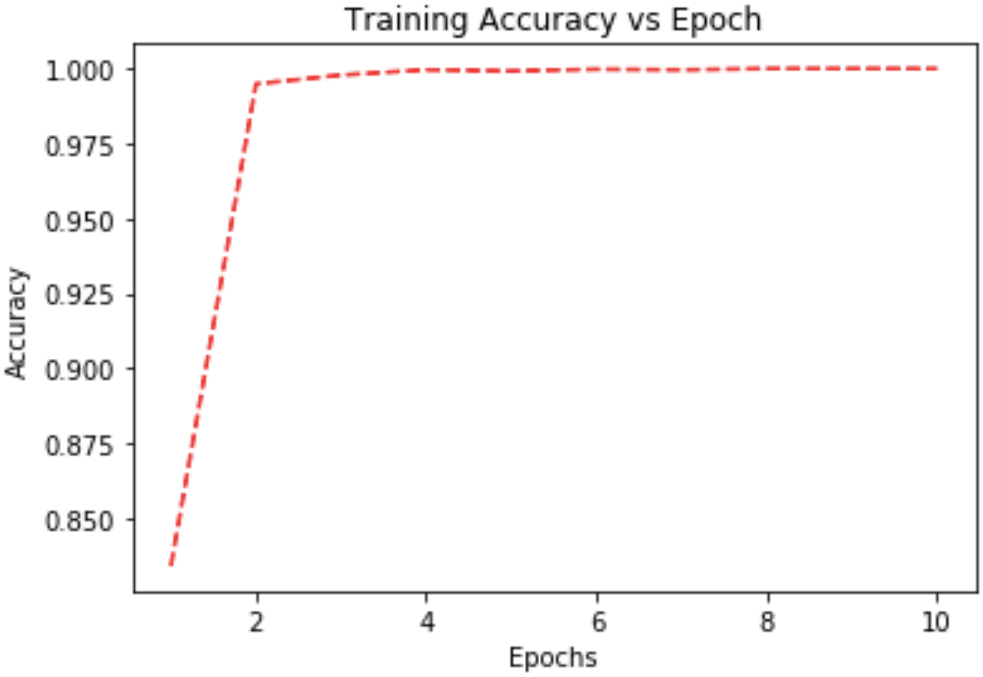


Figure 19. Training Accuracy vs Epochs

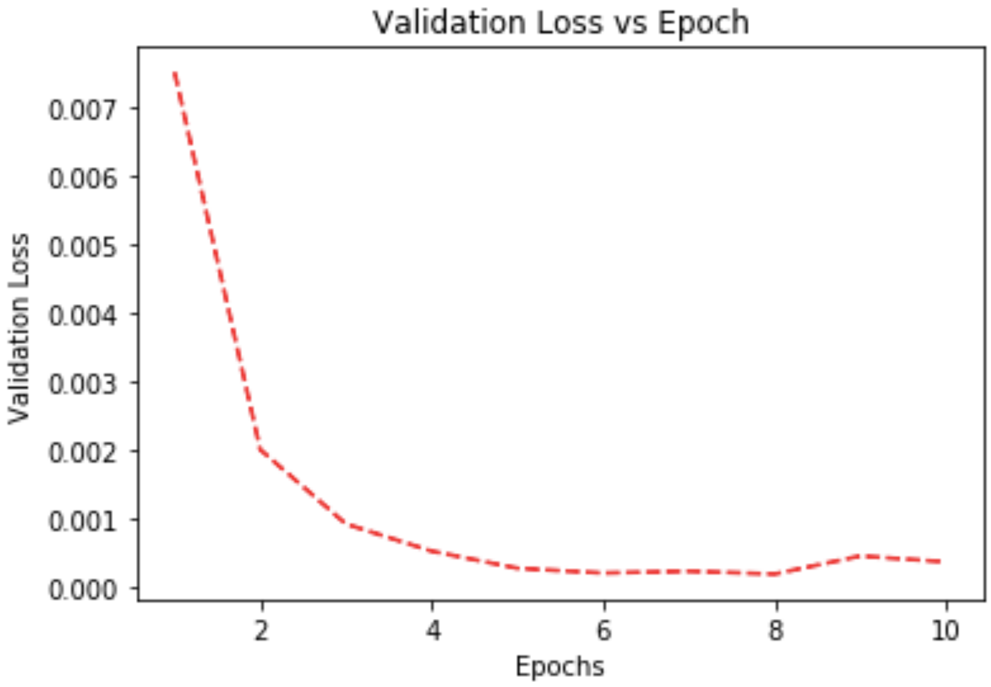


Figure 20. Validation Loss vs Epochs

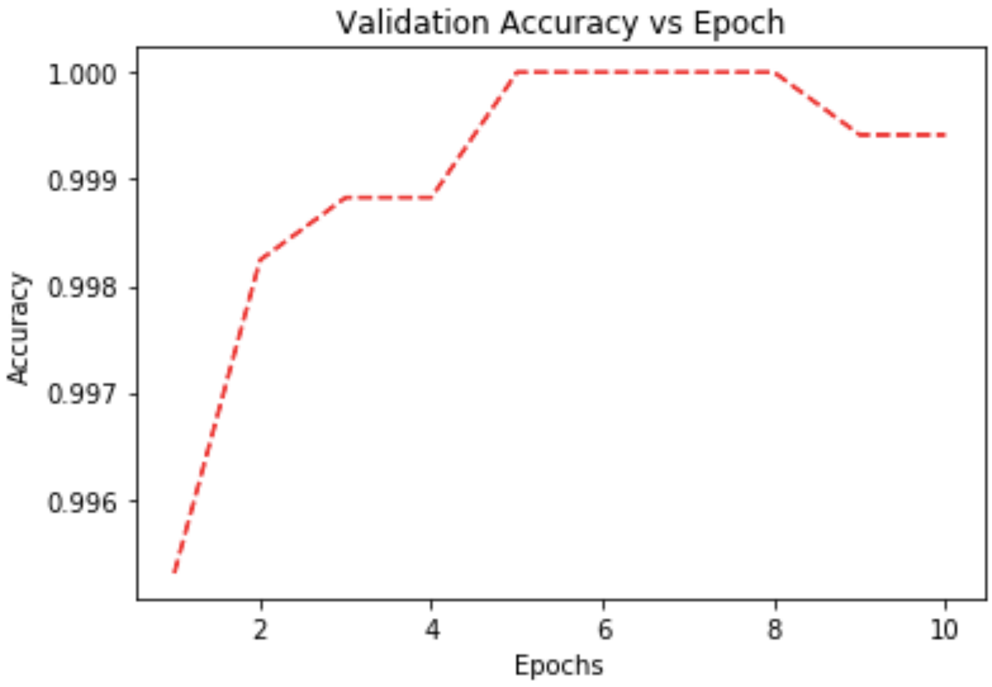


Figure 21. Validation Accuracy vs Epochs

Table I contains the ROC-AUC values for the models presented in this project as well as an existing tool, CFD, when tested on the CRISPOR dataset.

TABLE 1 - MEAN ROC-AUC VALUES

Model	**Mean ROC-AUC**
Support Vector Machine	0.85
Logistic Regression	0.90
Convolutional Neural Networks	0.90
Cutting Frequency Determination (CFD)	0.92

As evident from Table I, CFD performs the best on the CRISPOR dataset. However, Convolutional Neural Networks and Logistic Regression give comparable performances.

CHAPTER 8 - CONCLUSION

This project focused on classifying off-target sites for single guide RNAs and a target genome and calculating off-target scores for the guide RNAs using the probabilities obtained from the models. For the purpose of this project, only the Cas9 nuclease is considered. Three models were considered - Support Vector Machines, Logistic Regression and Convolutional Neural Networks. From the results, we can see that Convolutional Neural Networks and Logistic Regression achieve the highest accuracies, comparable to the existing tool CFD.

For future work, the models proposed in this project can be extended to accommodate other CRISPR systems such as Cpf1 [20]. The solution could also be modified to consider structural and physical features of the sequences as well, in addition to the existing set of features.

REFERENCES

[1] J. A. Doudna and E. Charpentier, “The new frontier of genome engineering with CRISPR-Cas9,” *Science*, vol. 346, no. 6213, pp. 1258096–1258096, Nov. 2014.

[2] L. A. Marraffini, “The CRISPR-Cas system of Streptococcus pyogenes: function and applications,” p. 17.

[3] P. D. Donohoue, R. Barrangou, and A. P. May, “Advances in Industrial Biotechnology Using CRISPR-Cas Systems,” *Trends Biotechnol.*, vol. 36, no. 2, pp. 134–146, Feb. 2018.

[4] P. D. Hsu, E. S. Lander, and F. Zhang, “Development and Applications of CRISPR-Cas9 for Genome Engineering,” *Cell*, vol. 157, no. 6, pp. 1262–1278, Jun. 2014.

[5] “CRISPR-Cas9 - gRNA design.” [Online].

[6] “An Intro to CRISPR-Cas9.” [Online].

[7] J. Listgarten *et al.*, “Prediction of off-target activities for the end-to-end design of CRISPR guide RNAs,” *Nat. Biomed. Eng.*, vol. 2, no. 1, pp. 38–47, Jan. 2018.

[8] G. Chuai *et al.*, “DeepCRISPR: optimized CRISPR guide RNA design by deep learning,”

[9] J. G. Doench *et al.*, “Optimized sgRNA design to maximize activity and minimize off-target effects of CRISPR-Cas9,” *Nat. Biotechnol.*, vol. 34, no. 2, pp. 184–191, Feb. 2016.

[10] M. Haeussler *et al.*, “Evaluation of off-target and on-target scoring algorithms and integration into the guide RNA selection tool CRISPOR,” *Genome Biol.*, vol. 17, no. 1, Dec. 2016.

[11] T. G. Montague, J. M. Cruz, J. A. Gagnon, G. M. Church, and E. Valen, “CHOPCHOP: a CRISPR/Cas9 and TALEN web tool for genome editing,” *Nucleic Acids Res.*, vol. 42, no. W1, pp. W401–W407, Jul. 2014.

[12] S. Q. Tsai *et al.*, "GUIDE-seq enables genome-wide profiling of off-target cleavage by CRISPR-Cas nucleases," *Nat. Biotechnol.*, vol. 33, no. 2, pp. 187–197, Mar. 2015.

[13] B. Zetsche *et al.*, "Cpf1 Is a Single RNA-Guided Endonuclease of a Class 2 CRISPR-Cas System," *Cell*, vol. 163, no. 3, pp. 759–771, Oct. 2015.

[14] "Support-vector machine," *Wikipedia*. 14-Mar-2019 [Online].

[15] "Logistic regression," *Wikipedia*. 12-Apr-2019 [Online].

[16] S. Swaminathan, "Logistic Regression — Detailed Overview," *Towards Data Science*, 15-Mar-2018. [Online]. Available: https://towardsdatascience.com/logistic-regression-detailed-overview-46c4da4303bc. [Accessed: 19-Apr-2019].

[17] "Sigmoid function," *Wikipedia*. 22-Feb-2019 [Online].

[18] "Convolutional neural network," *Wikipedia*. 09-Apr-2019 [Online].

[19] D. Liu, "A Practical Guide to ReLU," *TinyMind*, 30-Nov-2017 [Online]. .

[20] "Home - Keras Documentation." [Online]. Available: https://keras.io/. [Accessed: 24-Apr-2019].