UNIVERSITY OF CALIFORNIA

Los Angeles

Designing single guide RNA for CIRSPR-Cas9 base editor by deep learning

A thesis submitted in partial satisfaction

of the requirements for the degree

Master of Science in Statistics

by

Liangke Gou

2019

ABSTRACT OF THE THESIS

Designing single guide RNA for CIRSPR-Cas9 base editor by deep learning

by

Liangke Gou

Master of Science in Statistics

University of California, Los Angeles, 2019

Professor Ying Nian Wu, Chair

The CRISPR base editors are programmable DNA editing systems that induce single-nucleotide changes in the DNA using a fusion protein containing a catalytically defective Cas9, a cytidine or adenine deaminase, and an inhibitor of base excision repair. This genome editing approach has the advantage that it does not require the generation of double-stranded DNA breaks or a donor DNA template. The single guide RNAs (sgRNAs) enables the precise editing at the designed region using CRISPR base editor. Different sgRNAs have largely different efficacy for base editing, and computational prediction can facilitate the optimized design of sgRNAs with high editing efficacy, sensitivity and specificity. Here we present a convolutional neural network-based approach to predict the sgRNAs editing efficiency for CRISPR base editing. Firstly, we designed a large-scale sgRNA library of over 7,000 sgRNAs to introduce pre-mature stop codons into essential genes in yeast, where the yeast would drop out from the population if the sgRNA works efficiently due to the disruption of essential genes. The base editing efficiency of the 7,000 sgRNAs was measured by the log ratio change of sgRNA abundance at 72 h after the editing induction and at the beginning. We built a CNN model using the sgRNA sequence and the surrounding DNA context as the training input to predict the editing efficacy of any given sgRNA sequences. With architecture and parameter tuning, the CNN model surpassed the machine learning approaches tested. In addition, the CNN model fully automated the identification of sequences that may affect sgRNA editing efficacy in a data-driven manner.

To Mom and Dad

and Zijun

TABLE OF CONTENTS

Chapter 1 Introduction ***1***

CRISPR-Cas9 genome editing **1**

CRISPR-Cas9 base editor **2**

Computational approaches for efficient sgRNA designing **4**

Chapter 2 Materials and Methods ***6***

Data collection and processing **6**

Sequence one hot encoding **7**

K-mer feature extraction **7**

Convolutional neural network architecture and parameters **8**

Feature identification by deriving feature saliency map **8**

Machine-learning methods used for model comparison **9**

Logistic Regression 9

Naïve Bayes Classification 10

Support Vector Machine 11

K-Nearest Neighbors 13

Linear Discriminant Analysis 14

Chapter 3 Results ***15***

Genetic screens with the base editor PTC library **15**

CNN architectures selection **18**

Comparison of CNN model with machine learning approaches **21**

Large improvement space for CNN model with increasing training data **23**

Automating feature identification using saliency map 24

Chapter 4 Discussion.. 27

References... 30

LIST OF FIGURES

Figure 1.1. CRISPR-Cas9 base editor mediates specific sgRNA-programmed C to T conversion. 3

Figure 3.1 Measure the sgRNA editing efficiency by introducing pre-mature stop codon into essential genes. 16

Figure 3.2 sgRNA abundance at different time points after CRISPR base editing. 17

Figure 3.3 The correlation of sgRNA abundance between different time point pairs. 18

Figure 3.4 Prediction AUC of different CNN architectures. 19

Figure 3.5 The architecture of used CNN model for sgRNA efficiency prediction. 20

Figure 3.6 Model accuracy and loss during CNN training. 21

Figure 3.7 CNN model outperformed conventional machine learning for the task of predicting base editor on-target editing efficacy. 23

Figure 3.8 Cross-validation of editing efficacy prediction models trained on different size data sets. 24

Figure 3.9 Feature saliency map for three example sgRNAs editing efficiency. 26

ACKNOWLEDGEMENTS

First, I would like to express my unwavering gratitude to my thesis advisor, Yingnian Wu, who has been extremely supportive to my research and career development. For all three classes I took with Dr. Wu, he worked hard to make every knowledge point easy for understanding to students. I would find Statistics and Math way harder to learn without taking his classes. His advice and support paved the way to my future career.

I owe a special thanks to my husband, Zijun Zhang, for his encouragement, support and love. He is always at my side and gave me endless support during all the hard times and he has devoted wholeheartedly to our family. I could only imagine how difficult it would be without him.

I would like to express my great appreciation for my committee members: Jingyi Jessica Li and Janet Sinsheimer. I was lucky to have them on my committee and I owe them a debt of gratitude.

Last but not least, I would like to thank my parents for their unconditioned love and endless support during my life. I could not imagine the life without the support from my parents.

Chapter 1 Introduction

CRISPR-Cas9 genome editing

The ability to precisely and efficiently edit DNA sequences within the genome of living cells have been a major goal of the life sciences for centuries [1]. In recent years, a genome engineering technique via the CRISPR (clustered regularly interspaced short palindromic repeat)/Cas9 system has revolutionized biology, therapeutics, and biotechnology by allowing genetic material to be added, removed or altered at particular locations in the genome. CRISPR/Cas9 is an adaptive immune system used to protect bacterial and archaeal species against invasion by foreign DNA and phages [2][3]. The bacteria capture small pieces of DNA from the invaded viruses and use them to create DNA segments know as CRISPR arrays. The CRISPR arrays function as the memory immune cells to allow bacteria to remember the same viruses or the viruses with closely related genomes. If the viruses attack again, the bacteria produce RNA segments from the CRISPR arrays to target the viruses' DNA and then an endogenous enzyme Cas9 (CRISPR-associated protein 9) will be recruited to the targeting site to cut the viruses' DNA. Cas9 uses CRISPR sequences as a guide to recognize and cleave specific strands of DNA that are complementary to the CRISPR sequences. Cas9 activity depends on the presence of a protospacer adjacent motif (PAM) sequence in the target DNA, thereby enabling the CRISPR/Cas9 machinery to recognize self from non-self DNA and can specifically targeting a designed genomic region.

The CRISPR-Cas9 system works similarly in the lab where researchers create a small piece of RNA with a short "guide" sequence that binds to a specific target sequence of DNA in a genome. The small piece of RNA designed for genome editing is called single guide RNA (sgRNA or gRNA), and it contains a region of 20 nucleotide base pairs that complement the targeting region and the Cas9 enzyme cuts the DNA at the targeted location. Once the DNA is cut, researchers use the cell's own DNA repair machinery to add or delete pieces of genetic material, or to make changes to the DNA by replacing an existing segment with a customized DNA sequence.

CRISPR-Cas9 base editor

Scientists have invented a catalytically inactive version of Cas9 by introducing multiple mutations, and this version was called dead Cas9 or dCas9. Dead Cas9 can be used as a DNA targeting tool to tether different enzymatic activities to specific DNA sequences, resulting gene repression, activation and epigenome editing. A specific base editor is invented by fusing a catalytically dead dCas9 to a cytidine deaminase protein, and it can alter DNA bases without inducing a DNA break. Base editors convert C->T (or G->A on the opposite strand) within a small editing window specified by the sgRNA [4] (Figure 1.1).

Base editing is a new genome editing technology that enables the direct, irreversible conversion of a specific DNA base into another at a targeted genomic locus. Importantly, this can be achieved without requiring double-stranded DNA breaks (DSB) given the inactivation of the Cas9 endonuclease. Since many genetic diseases arise from point mutations, this technology has important implications in the study of human health and disease.

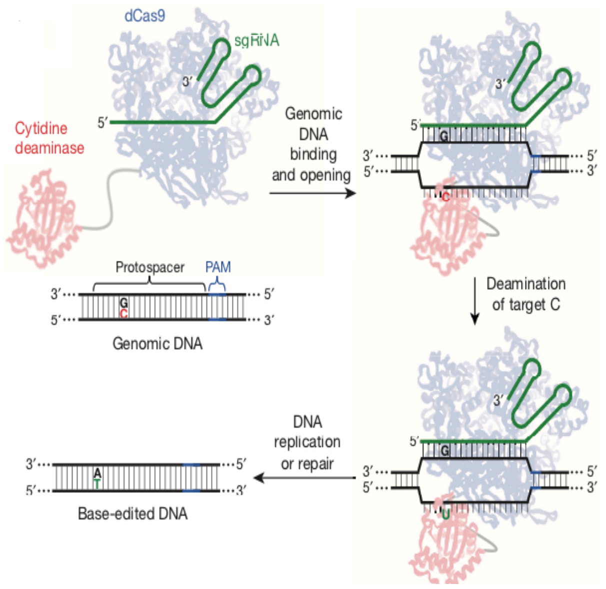

Figure 1.1. CRISPR-Cas9 base editor mediates specific sgRNA-programmed C to T conversion.

The sg RNA (green) is designed to complement a position of interest in the genomic DNA. The sgRNA can recruit the dCas9 to the targeting region. The dCas9 is fused with a Cytidine deaminase APOBEC1 enzyme (red), which can specifically convert the C at the targeting region into U. The resulting G:U heteroduplex can be permanently converted to an A:T base pair following DNA replication or DNA repair. Therefore the process achieved the C to T conversions. This figure is for background illustration and is from Komor et al., Nature, 2016 [4].

Computational approaches for efficient sgRNA designing

The efficiency of the base editing varies across the genome, mainly depending on the sequence surrounding the target nucleotide C, and the location of the target C within the sgRNA sequences. Designing the sgRNAs that will enable highly efficient and specific base editing is essential when using base editing. Given base editing was invented in three years ago, very few computational approaches have been developed for designing base editor sgRNA, whereas various sgRNA design rules and tools have been developed for sgRNA on-target identification and efficacy prediction for original Cas9 to generate double strand breaks, where the location of the target C was not taken into account. Understanding the methods for Cas9 on-target efficiency prediction can help us design approaches for base editor sgRNA, and those approaches for original Cas9 sgRNA design can be characterized in four types: (1) alignment-based. These types of methods design sgRNAs by considering the position of PAM and the off-target sites [5]; (2) hypothesis-driven, where sgRNA efficacy was scored by genome context and the off-target effects analyzed by Bowtie2 [6]; (3) learning-based method using data from a designed screening experiment. Deonch et al. used the data from a designed screening library and large-scale assessment of on- and off-target activity to improved algorithms for sgRNA efficiency prediction [7]; (4) deep-learning based methods that use sgRNA sequence to predict efficacy. Chuai et al. developed a framework for sgRNA on-target and off-target sites prediction automated identification of sequence via deep learning [8].

So far to our knowledge, no one has applied deep learning to sgRNA efficiency prediction of CRISPR-Cas9 base editor. In recent years deep learning had shown its power in biological researches, such as inferring differential alternative splicing using low coverage RNA sequencing [9]. The success of deep neural networks in off target effect prediction in the original Cas9 system inspires us to extend the applications to sgRNA efficiency prediction in CRISPR-Cas9 base editor [10]. Here I took advantage of deep learning and developed a convolutional neural networks (CNN) model to predict the efficiency of sgRNAs. Our CNN model has surpassed several machine-learning methods tested, and has the advantage of fully automated feature identification. We also gained biological meaningful interpretation of the model using feature saliency maps.

Chapter 2 Materials and Methods

Data collection and processing

The data used in this study are the original data generated from a large-scale sgRNA library screening for CRISPR base editors by our lab. We have synthesized 7,000 sgRNAs, where 6,000 sgRNAs targeted and introduced pre-mature stop codons to the essential genes in yeast. The other 1,000 sgRNAs were used as the control by targeting non-essential genes or non-yeast genome. If the sgRNA efficiently functions for base editing, the targeted essential gene will be terminated, causing death to that yeast. The death of the yeast cell will lead to the drop out of that sgRNA. On the other hand, the less efficient sgRNAs cannot provide efficient disturbance to the essential genes, and yeast cells will survive healthily. 7,000 sgRNAs were transformed into a pool of millions of yeast with the CRISPR-Cas9 base editor under a galactose inducible promoter Gal. Base editor was turned on at timepoint 0. Around one million yeast were sampled out and their DNA was extracted for sequencing at six different time points: 0 hr, 24 hrs, 36 hrs, 48 hrs, 60 hrs and 72 hrs after turning on base editor. The abundance for each sgRNA was used as an indicator for the efficacy of that sgRNA. The log ratio between 72 hrs and 0 hr sgRNA abundance was used as the training and testing data. We binarized the data using 0 as the threshold for the log ratio: for log ratio larger than 0, we assign a value of 0 as this indicates the less efficient sgRNA, otherwise we assign a value of 1.

Sequence one hot encoding

The original sgRNA sequences were made up with A, G, C, T and each base in the sgRNA can be encoded as one of the four one-hot vectors [1, 0, 0, 0], [0, 1, 0, 0], [0, 0, 1, 0] and [0, 0, 0, 1]. The sgRNA sequence is 20 base pairs (bp), and we also incorporate the upstream and downstream 5 bp to the input to represent the DNA context. In this way, every input sequence is 30 bp.

Using one hot encoding, every sgRNA-plus-DNA-context sequence can be represented by a 4×30 matrix where 30 is the length of the sequence which includes the upstream and downstream 5 bp to the 20 bp gRNA region. The code matrix of sgRNA will be directly fed into CNN-based models for training and testing.

K-mer feature extraction

K-mer features extracted from the same 30 bp DNA sequences that contains sgRNA and upstream and downstream 5 bp were used for machine learning approaches. A k-mer analysis of a DNA sequence is considered as an extraction which counts of every DNA word within length k (k bases along one strand) using a 'sliding window' approach to eliminate the influence of an arbitrary chosen starting point. Here we extracted the information of length $1 \leq k \leq 4$ within every sgRNA sequence. We chose $k = 4$ as the upper bound for k because this would generate a decent number of features for modeling without created large computational cost. Altogether 340 features were generated using the 1-mer to 4-mer features extracted from the DNA sequences.

Convolutional neural network architecture and parameters

Figure 3.5 and the following description gives a summary of the basic architectural structure of CNN used: the input is a code matrix with shape of 30 (sequence length of sgRNA and upstream and downstream 5-bp) $\times$ 4 (size of nucleotides vocabulary).

The first layer of our network is a convolutional layer, which is designed for extracting sgRNA matching information using 32 filters of 3 different sizes. Thus, it gives a 1 $\times$ 30 $\times$ 32 feature map from this layer. The second layer is also a convolutional layer with 32 filters of 3 different sizes. It has the feature map layout as the first layer.

The third layer is a global average-pooling layer connected with the previous convolutional layer. Each of these pooling windows outputs the average value of all of its respective convolutional layer outputs. The size of each pooling window used in the CNN is 2; other sizes are also tested in the following experiments. Accordingly, it gives a 1 $\times$ 2 $\times$ 32 feature map to the next layers. The pooling layers progressively reduce the spatial size of the representation to reduce the amount of parameters and computation in the network. The following layers are a flattern layer and two fully connected dense layers.

Feature identification by deriving feature saliency map

We generated positive class-specific saliency maps for sgRNAs that work efficiently. Saliency maps were initially developed in the computer vision field. The idea of saliency is to compute the gradient of output category with respect to input. This represents how output category value changes with respect to a small change in input. All positive values in the gradients indicate a small change to the input increase the output value. Visualizing these gradients should provide intuition of feature importance.

Machine-learning methods used for model comparison

The readout from the experiments was binarized into 1 (working) and 0 (not working) through one hot encoding, turning the prediction task into a classification problem. Here we applied five different machine-learning methods to predict the sgRNA editing efficiency based on the k-mer features extracted from the sgRNA DNA sequence and the 5 bp upstream and downstream of the sgRNA. In the following, we briefly discussed the background of the five machine learning methods applied:

Logistic Regression

Logistic regression can be considered as a machine-learning algorithm that is used for the classification problems. As far as machine learning is concerned, it is a predictive analysis algorithm and in statistics is a generalized linear model with a logit link function. The link function of logistic regression is the logistic function, as shown in equation (1). Here in our base editor dataset, we had around 5400 training sgRNA sequences with 340 k-mer features extracted from each sgRNA sequence. We built a feature matrix X with x_i^T=($x_{i1}, x_{i2}, \dots, x_{ip}$), where i represented the i th training sgRNA sequnence and p standed for the p th feature for that sequence. We defined $y_i \in \{0,1\}$ as the outcome working or not working label for the sgRNA editing efficiency. If we assume $[y_i|x_i, \beta] \sim Bernoulli(\theta_i)$ and

$$logit(p_i) = log\frac{\theta_i}{1-\theta_i} = s_i = {x_i}^T\beta \tag{1}$$

Then

$$\theta_i = sigmoid(s_i) = \frac{e^{s_i}}{1+e^{s_i}} = \frac{1}{1+e^{-s_i}} \tag{2}$$

where the sigmoid function in the inverse of the logit function.

Naïve Bayes Classification

In the naïve Bayes framework, a classification problem aims to find the outcome with maximum probability given a set of observed variables. The classification was based on the Bayes rule, where

$$P(A|B) = \frac{P(B|A)P(A)}{P(B)} \tag{3}$$

Given a sgRNA sequence example, described by its k-mer feature vector $X = (x_1, x_2, \dots, x_n)$, we are looking for a label Y that maximizes the likelihood $P(X|Y) = P(x_1, x_2, \dots, x_n|Y)$, where Y is the random variable for label. Since the current work is to predict the sgRNA working or not working, a binary class $Y \in \{0,1\}$ was generated, where 1 denotes that the sgRNA works and 0 denotes the sgRNA sequence does not work. For the binary classification, the class for the sgRNA function efficiently or not could be determined by comparing two posteriors as

$$\frac{P(Y=1|x_1, x_2, \dots, x_p)}{P(Y=0|x_1, x_2, \dots, x_p)} = \frac{P(Y=1)\prod_{i=1}^{n} P(x_i|Y=1)}{P(Y=0)\prod_{i=1}^{n} P(x_i|Y=0)} \tag{4}$$

Taking the logarithm of equation above, we obtain

$$\log\frac{P(Y=1|x_1,x_2,\dots,x_p)}{P(Y=0|x_1,x_2,\dots,x_p)} = \log\frac{P(Y=1)}{P(Y=0)} + \sum_{i=1}^{n}\log\frac{P(x_i|Y=1)}{P(x_i|Y=0)} \tag{5}$$

$\log\frac{P(Y=1)}{P(Y=0)}$ is a constant, so $\log\frac{P(Y=1|x_1,x_2,\dots,x_p)}{P(Y=0|x_1,x_2,\dots,x_p)} - \log\frac{P(Y=1)}{P(Y=0)}$ is the log Bayes factor. If the log of the Bayes factor is positive, the likelihood ratio has increased, meaning the sgRNA is more likely to be an efficient sgRNA ($Y=1$) [11].

Equation (4) made the assumption that the features were independent. In reality, the extracted k-mer features were not exactly independent, for example if you observed a 2-mer ‘AA’, then the probability of seeing a ‘AAA’ 3-mer will be larger. The correlation between features may decrease the prediction power of the model, but Naïve Bayes Classification is still a powerful model.

Support Vector Machine

Support Vector Machine (SVM) is a discriminative classifier formally defined by a separating hyperplane. Given the labeled training data, the algorithm will output a hyperplane that maximize the margins between samples of two classes, which could be used for categorizing new examples. In two-dimensional space, the hyperplane is a line dividing the space into two parts where each class lay in either side.

The maximum-margin classifier is the discriminant function that maximizes the geometric margin 1/||w||, which is equivalent to minimizing $||w||^2$. This leads to the following constrained optimization problem:

$$minimize\ \frac{1}{2}\ ||w||^2 \tag{6}$$

$$\mathit{subject\ to}: \quad y_i(w^T x_i + b) \geq 1 \quad i = 1, \dots, n$$

In equation (6), X is the feature matrix of training individuals, where x_i stands for the features of the i th sgRNA sequence. y_i is the label of that corresponding sgRNA. w is a set of weights (one for each feature) whose linear combination predicts the value of y. Both w and b needs to be solved during SVM training. The constraints in this formulation ensure that the maximum-margin classifier classifies each example correctly, which is possible since we assumed that the data are linearly separable. In practice, data are often not linearly separable; and even if they are, a greater margin can be achieved by allowing the classifier to misclassify some points. To allow errors we replace the inequality constraints in the previous equation with

$$y_i(w^T x_i + b) \geq 1 - \xi_i \quad i = 1, \dots, n$$

where ξ_i are slack variables that allow an example to be in the margin (1≥ξi≥0, also called a margin error) or misclassified (ξi≥1). The optimization problem now becomes

$$\mathit{minimize}\ \frac{1}{2}\ ||w||^2 + C \sum_i \xi_i \tag{7}$$

$$\mathit{subject\ to}: \quad y_i(w^T x_i + b) \geq 1 - \xi_i \quad \xi_i \geq 0$$

The constant $C > 0$ sets the relative importance of maximizing the margin and minimizing the amount of slack. This formulation is called the soft-margin SVM and was introduced by Cortes and Vapnik [12][13]. We applied SVM to our base editing database, where X is the feature matrix and Y is the label variable.

K-Nearest Neighbors

K-Nearest Neighbors (KNN) classifier is to classify unlabeled observations by assigning them to the class of the most similar labeled examples. The algorithm calculates the distance between the unlabeled observations and other labeled training samples. By default the KNN function employs Euclidean distance, which can be calculated with the following equation

$$D(x_i, x_j) = \sqrt{\left(x_{i1}-x_{j1}\right)^2 + \left(x_{i2}-x_{j2}\right)^2 + \cdots + \left(x_{ip}-x_{jp}\right)^2} \tag{8}$$

where x_i and x_j are subjects to be the i th and j th sgRNA sequences with p features [14]. There are also other methods to calculate distance such as Manhattan distance [15]. The same as descripted in previous sections, X represents the feature matrix, and Y is the working or not working label of sgRNAs. The features p used in this analysis is the k-mer features extracted from sgRNA sequences.

In the classification phase, the unlabeled sgRNA (can be seen as a query point) is classified by assigning the label which is most frequent among the k training samples nearest to the query point. The appropriate choice of the parameter k, which decides how many neighbors will be chosen for KNN algorithm, has significant impact on the performance of KNN algorithm. Choosing an appropriate k value strikes a balance between overfitting and underfitting [16], and this process can be tuned through cross validations. Some authors suggest setting k equal to the square root of the number of observations in the training dataset [17]. I had ~5400 sgRNAs training date, and the square root of the training size was 73.5. I used k equals to 100 for the KNN analysis in this study.

Linear Discriminant Analysis

Linear Discriminant Analysis (LDA) is also a technique people used for classification problems. In the traditional sampling scenario, LDA is employed in a univariate setting. Let us assume a set of $n = n_0 + n_1$ independent sample points, where $X_1, X_2, \dots, X_{n_0}$ come from population $\prod_0$ and $X_{n_0+1}, X_{n_0+2}, \dots, X_{n_0+n_1}$ come from population $\prod_1$. In our dataset, we used $\prod_0$ as the group of less efficient sgRNAs and $\prod_1$ as the efficiently working sgRNAs, and X_i is the k-mer feature vector for the i th sgRNA sequencing. Population $\prod_i$ is assumed to follow a univariate Gaussian distribution $N(\mu_i, {\sigma_i}^2)$ for $i = 0{,}1$. LDA utilizes the Anderson W statistic, which in the univariate case is presented as

$$W(\bar{X}^0, \bar{X}^1, X) = \frac{1}{\hat{\sigma}^2}\left(X - \frac{\bar{X}^0 + \bar{X}^1}{2}\right)^T (\bar{X}^0 - \bar{X}^1) \tag{9}$$

Where $\bar{X}^0 = \frac{1}{n_0}\sum_{i=1}^{n_0} X_i$ and $\bar{X}^1 = \frac{1}{n_1}\sum_{i=n_0+1}^{n_0+n_1} X_i$ are the sample means for each class and $\hat{\sigma}^2$ is the pooled estimate of the variance of classes, which is assumed to be common in the LDA discriminant. Given $\bar{X}^0$ and $\bar{X}^1$, the designed LDA classifier is given by

$$\psi(X) = \begin{cases} 1, if\ W(\bar{X}^0, \bar{X}^1, X)\ \le c \\ 0, if\ W(\bar{X}^0, \bar{X}^1, X) > c \end{cases} \tag{10}$$

with c being a constant. It is commonly assumed that c is zero.

Chapter 3 Results

Genetic screens with the base editor PTC library

We used large-scale oligonucleotide synthesis to generate a pool of more than 7,000 distinct sgRNA plasmids to introduce premature termination codons (PTCs) to different sites in 1,034 yeast genes considered essential for viability. Each gene was targeted by multiple sgRNAs. We transformed haploid nej1Δ yeast in bulk with plasmid pools. We sequenced the sgRNAs from a sample of millions of the transformed cells before inducing Cas9 base editor, and using this sgRNA counts as the baseline sgRNA abundance at time point 0 h. After inducing Cas9 base editor expression, we collected millions of surviving transformed cells every 24 h for 3 days to sequence the barcodes for sgRNAs detection. sgRNAs that can efficiently disrupt the function of genes essential for viability were expected to drop out of the pool over time, whereas those that do not were expected to persist (Figure 3.1).

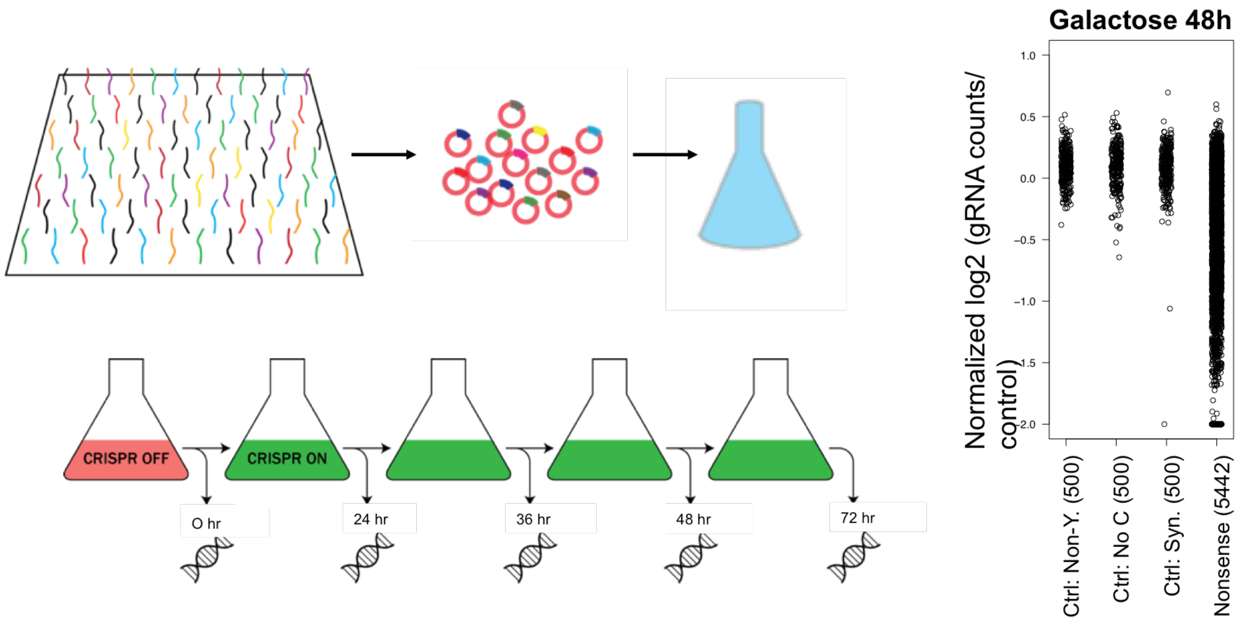

Figure 3.1 Measure the sgRNA editing efficiency by introducing pre-mature stop codon into essential genes.

The schema in Figure 3.1 shows the experimental design. A library pool of around 7,000 sgRNAs was transformed into yeast. After Cas9 base editor induction, DNA was extracted at different time points. At each time point, edit-directing plasmids were quantified by sequencing. 1,500 sgRNAs worked as the control sgRNAs: 500 sgRNAs targeting non-yeast genes, 500 sgRNAs targeting regions without a cysteine (C base pair), 500 sgRNAs introducing synonymous changes which won't change the protein amino acid sequence. The non-sense panel are those ~6,000 sgRNAs targeting the essential genes and introduced pre-mature stop codons. The effectively functioned sgRNAs dropped out and had low log ratio, while some of the sgRNAs not worked well with the similar log ratio as the control sgRNAs.

We calculated the log ratio between the sgRNA abundance at each time point compared to the starting baseline abundance at time point 0 h for all sgRNAs. The log ratio of the sgRNAs extracted at the time point 24 h, 36 h, 48 h, 60 h and 72 h were calculated and the distribution is shown in Figure 3.2.

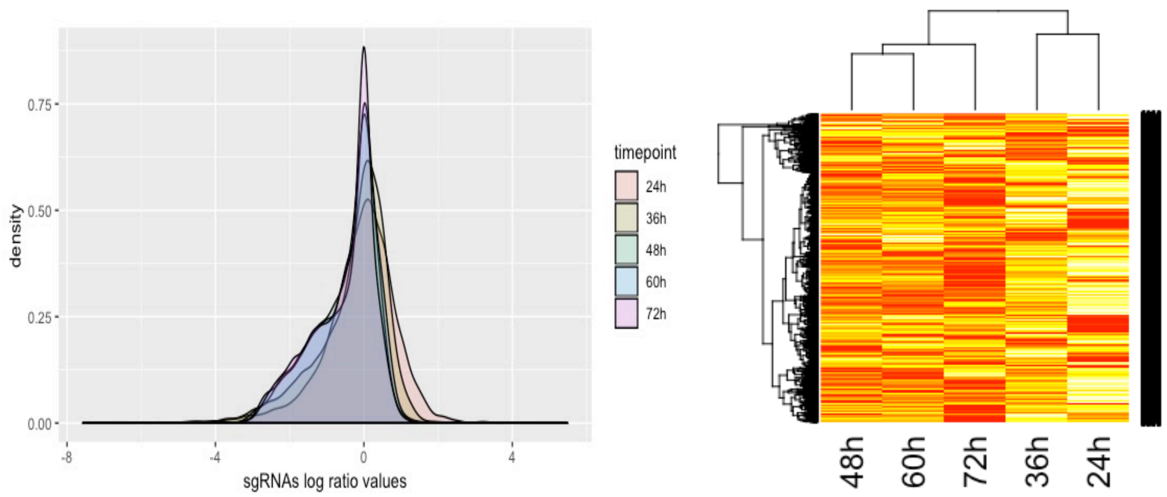

Figure 3.2 sgRNA abundance at different time points after CRISPR base editing.

Left panel: The density of sgRNA log ratios at different time points. Right panel: The log ratios of ~7,000 sgRNAs at five different time points were clustered using hierarchical clustering, where we can see 48 h, 60 h and 72 h clustered together.

Hierarchical clustering was performed with the data of these five time points, and the sgRNAs log ratios at time points 48 h, 60 h and 72 h were clustered more closely than the sgRNAs log ratios at time points 36 h or 24 h (Figure 3.2), which was consistent with the biological process that Cas9 base editor function gradually and stable editing may occur after 48 hours. Once editing occurred, the functional sgRNA would drop out and decrease the perturbation of sgRNAs abundance. We also measured the correlation between the sgRNA log ratios at every time point pairs. 48 h and 60 h after the Cas9 base editor induction had the highest correlation (0.838, SE =), followed by the correlation between 60 h and 72 h (0.831, SE=). The lowest correlation was observed for the sgRNA log ratios between 24 h and 72 h, which was expected given at 24 h the Cas9 base editor started actively working and it required time for sgRNAs to drop out; while at 72 h, the sgRNAs had already dropped out. In general, the log ratios at 48 h, 60 h and 72 h were highly correlated (Figure 3.3). Therefore, I used the sgRNA abundance ratio at 72 h after CRISPR base editor induction as the training data to develop the deep learning model that can predict the sgRNA editing efficiency based on the sgRNA sequence and the DNA contexts. The readout was binarized using 0 as the threshold, where the sgRNA log ratio less than 0 to be label 1, indicating functional worked sgRNAs.

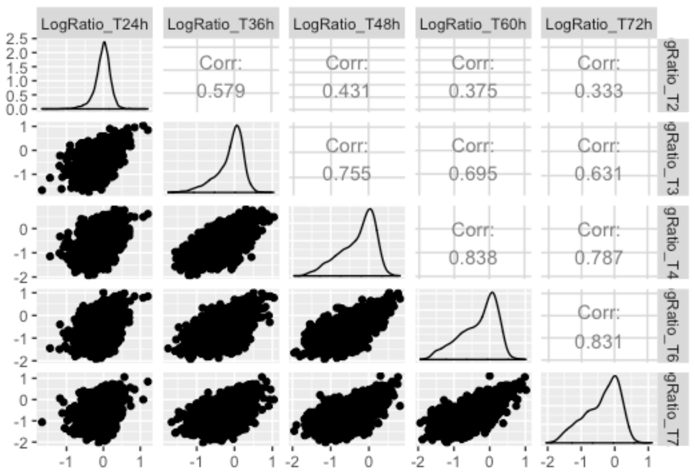

Figure 3.3 The correlation of sgRNA abundance between different time point pairs.

The sgRNAs log ratio at all time points were positively correlated. Diagonal showed the distribution of the sgRNA log ratio at different time points. The correlation between every two time points were labeled in the left upper corner.

CNN architectures selection

In order to gain the insight of a well-structured convolutional neural network (CNN) model, I compared the performance of different model architectures trained on the base editor sgRNA efficiency data. I first tried to compare how many layers of convolution performed better by comparing three architectures: one convolutional layer plus 1 pooling layer (1C1P), two convolutional layers plus 1 pooling layer (2C1P), two convolutional layers plus 1 pooling layer and two convolutional layers plus 1 pooling layer (2C1P2C1P). From the testing AUC the two convolutional layers plus 1 pooling layer (2C1P) performed best. Therefore, I used this layer structure and further tested the number of filters and filter size. Comparing the number of filters as 16, 32 and 64, we saw that 32 filters work best. The filter size 1,3 and 6 were tested and within these filter size 3 outperformed. Through the exploration and comparison of different architectures, I decided the CNN model used for later analysis to be two convolutional layers plus one pooling layer, with the number of filters 32 and filter size 3 (Figure 3.4).

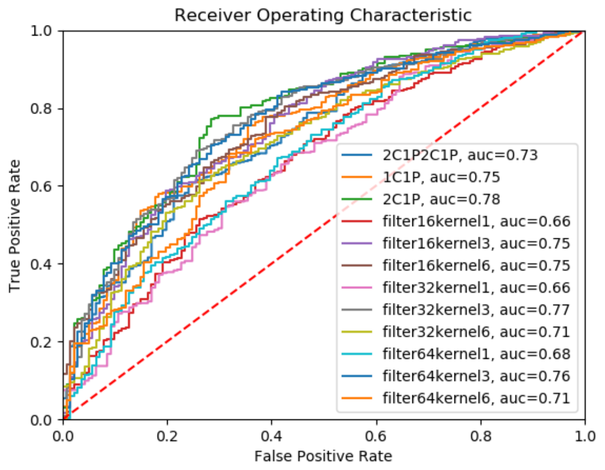

Figure 3.4 Prediction AUC of different CNN architectures.

Comparison of sgRNA editing efficacy predictions showed the two convolutional layers plus one pooling layer architecture performed better than other tested architectures.

From the model architecture selection and the tuning for the number of filters and filter sizes, an optimized model with two convolution layers, one pooling layer, one flatten layer and two dense layer stood out. For both convolution layers, I used 32 filters with the filter size 3. Average pooling with pooling size 2 was applied in the pooling layer, followed by two fully connected layers before the final output (Figure 3.5).

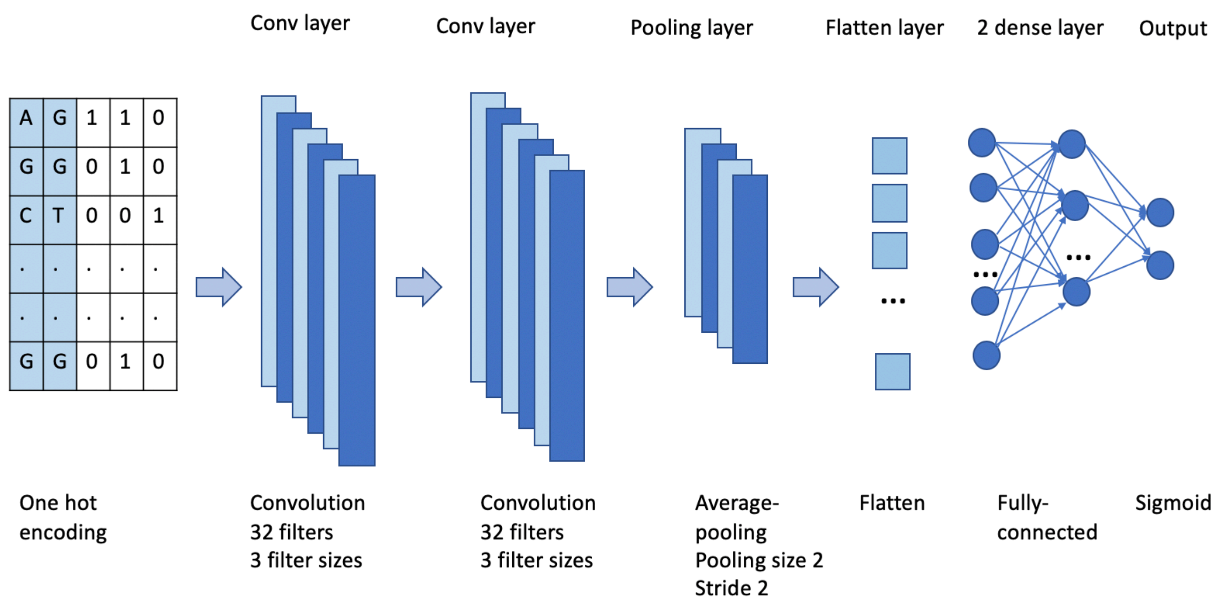

Figure 3.5 The architecture of used CNN model for sgRNA efficiency prediction.

The input of this deep neural network is the encoded sgRNA sequence with length 30. The model contains two convolutional layers, one pooling layer, flatten and two dense layers. The convolutional layers consist of 32 filers with filter size 3. The global average-pooling layer applies a filter with window size 2 to the previous layers. The outputs of average-pooling layer are joined together into one vector by flattening. Each neurons in the flatten layer is fully connected to the first dense layer. The neurons in output layer and dense layers use sigmoid function as the activation function, while all the neurons in other layers use ReLU as the activation function.

I applied the CNN structure with the selected parameters and architecture shown in Figure 3.5 to train the prediction model. With each training epoch, the training accuracy kept increasing and training loss continued decreasing. For the testing accuracy, we also observed a steady increase while training proceeded. Early stopping criteria reached after 29 epochs. The trend for the loss function showed the testing loss was not decreased after 25 epochs. The testing accuracy reached around 0.85 after 25 epochs (Figure 3.6).

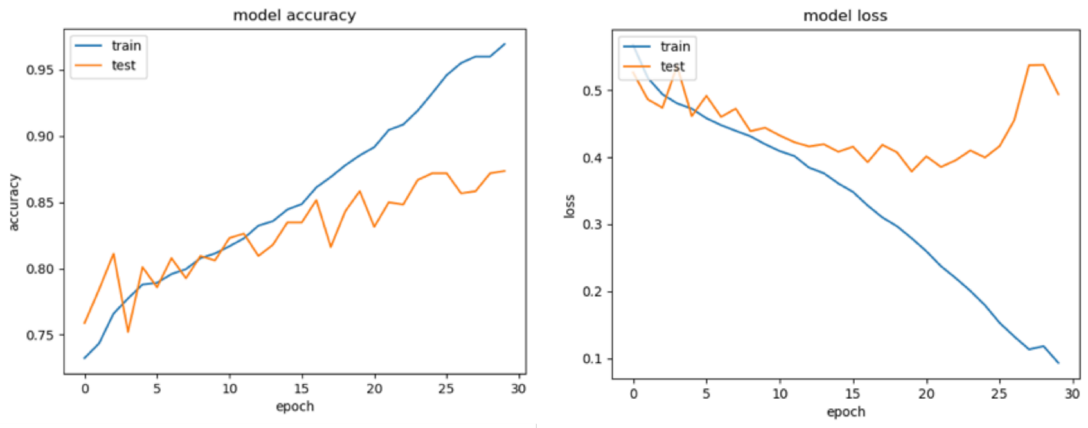

Figure 3.6 Model accuracy and loss during CNN training.

Loss in the testing group increased after 25 epochs and the CNN model reached the early stopping criteria after 29 epochs. The testing accuracy gained was above 0.85.

Comparison of CNN model with machine learning approaches

To evaluate the ability of CNN in sgRNA on-target efficacy prediction, we tested the prediction AUC for five different machine-learning approaches for the same sgRNA base editing efficacy. After excluded the 1,000 control sgRNAs, the whole dataset includes around 6,000 sgRNAs to introduce the pre-mature stop codon to around 1,043 experimentally validated known essential genes. Given I formulated the CNN model in a classification schema, I used the same binarized data and with the same classification problem for the machine learning methods for a rigorous comparison. As what was applied for the CNN model, the sgRNA target and work efficacy were labeled in a binary way according to the threshold 0 for the log fold change of the sgRNA abundance. K-mer features of K from 1 to 4 were generated as features to train the five different machine learning models. The AUC of each methods were calculated for all tested approaches (Figure 3.7). For a more robust AUC comparison, I calculated the average AUC from 10-fold cross-validation. CNN model had an average AUC from 10-fold cross-validation that is over 0.75, outperforming all other tested approaches. Followed by logistic regression and LDA, which have the average AUC around 0.65. Naïve Bayesian and SVM had similar AUC and have slightly lower AUC compared to logistic regression and LDA. KNN had the lowest average AUC in the comparison (Figure 3.7).

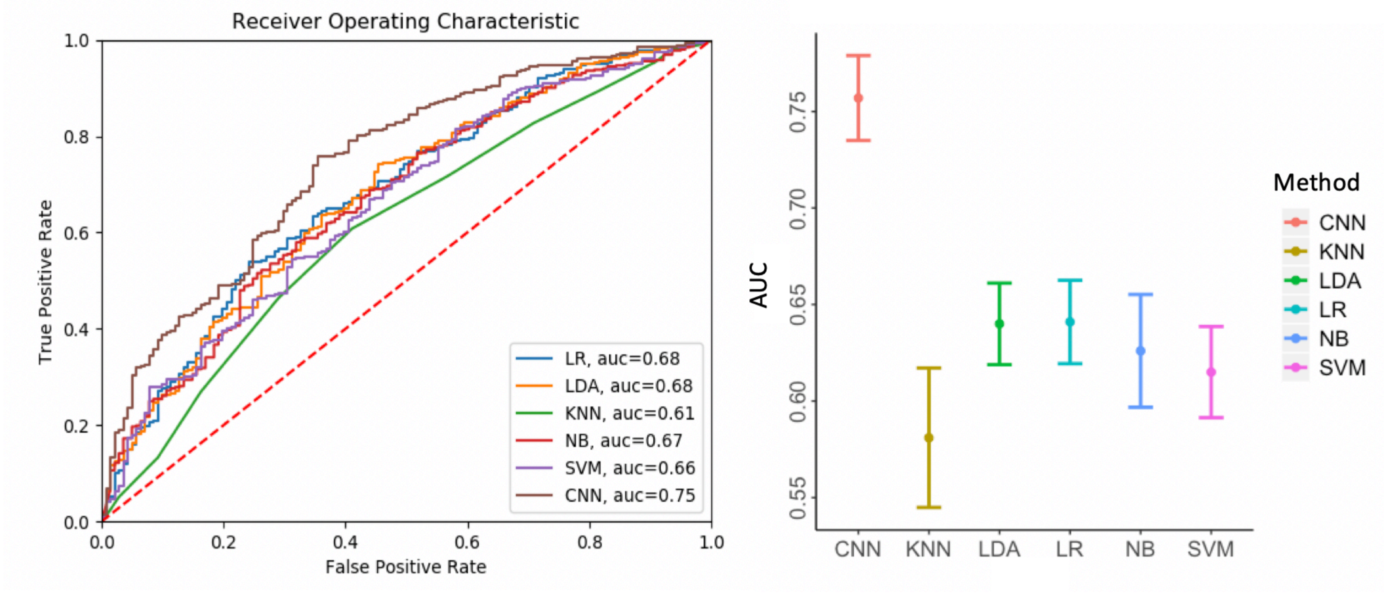

Figure 3.7 CNN model outperformed conventional machine learning for the task of predicting base editor on-target editing efficacy.

Left panel shows the AUC comparison of CNN and five machine learning methods, where CNN achieved the highest AUC. The right panel showed the average number of AUC from ten-fold cross validation for CNN and machine learning methods. CNN performed best within the tested models. 'LR' represents logistic regression; 'KNN' represents k-nearest neighbors; 'LDA' stands for linear discriminant analysis; 'NB' means naïve Bayesian and 'SVM' shows support vector machine.

Large improvement space for CNN model with increasing training data

We next evaluated the how the performances of different models changed when the sample size increased. As the size of training data for the cross-validation increased, the average AUC from 10-fold cross-validation steadily increased for all methods without reaching to a plateau. The trend suggests that with larger sample size in training data, a better AUC will be achieved, in particular for the CNN model (Figure 3.8). The homebrew training sample in my study is around 5,400 sgRNAs, partially explains why the best AUC we achieved so far is just more than 0.75.

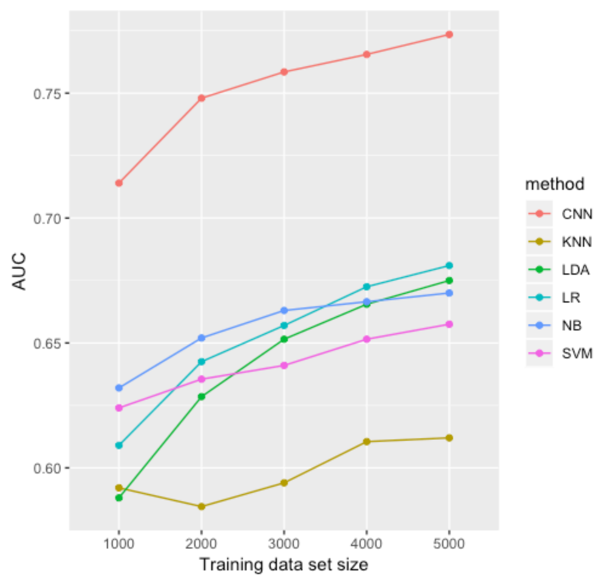

Figure 3.8 Cross-validation of editing efficacy prediction models trained on different size data sets.

The AUC performance for CNN and five machine learning methods was measured in a ten-fold cross validation with the sequence composition when the training data size was 1000, 2000, 3000, 4000 and 5000. Each point represents the average result of ten outer sets.

Automating feature identification using saliency map

Previously some arguments against deep learning in the genomics field pointed out the 'black box' feature of neural network. However, with the increasingly high-speed development in the field, the feature identification and visualization based on CNN learning models have been extensively addressed. We present the feature importance for efficient sgRNA design using the saliency map of the trained deep learning model. This allows automatically generate the feature importance showing which specific base pair plays important roles at which specific position.

We generated the feature saliency map for sgRNA editing efficacy prediction using the existing training data, and the map reflected how output category value changed with respect to a small change in the input. Interestingly, we obtained feature saliency maps consistent with previous findings from biological experiments. First of all, from biological knowledge, a cysteine (base pair C) between the region 4-8 is crucial for the base editor to function given the characteristic of the deaminase protein. From the saliency map, the cysteine at position 4 of examples sgRNAs all have a strong feature importance, entirely consistent with the knowledge (Figure 3.9). In addition, saliency map show that the bases at the front positions of sgRNAs matters more than the base pairs at the end (Figure 3.9). This also consistent with the property of base editor as the deaminase protein will only function on the beginning regions of sgRNAs. The learned saliency feature map provides us with the biological interpretable results and shows its potential for new biology discoveries.

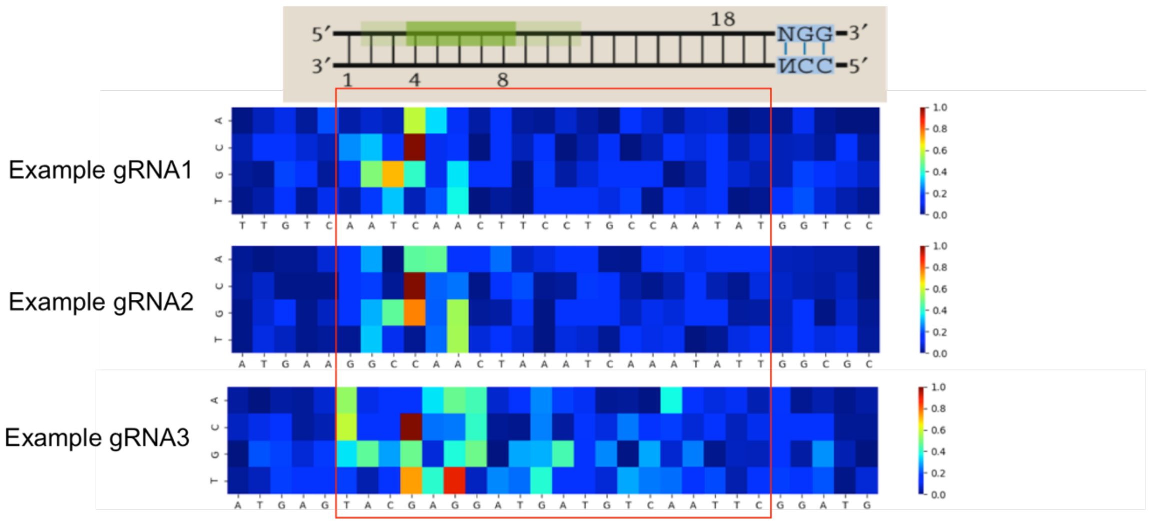

Figure 3.9 Feature saliency map for three example sgRNAs editing efficiency.

The top showed the essential position for a cysteine (C) on sgRNA based on previous experimental data in literature. The green part showed a cysteine (C) between positions 4-8 played important role for the base editor to work, which is consistent with the observation that the C at position 4 in these three examples had large feature importance.

Chapter 4 Discussion

The recent developed CRISPR-mediated base editing tools enable the direct conversion of one nucleotide to another without creating double strand breaks in the target sequence and without the introduction of donor DNA templates. The CRISPR base editors have wide applications in research and therapeutic fields. A well designed sgRNA enables the high efficiency and specificity of the editing of base editors. Here we developed an efficient convolutional neural network model for CRISPR-Cas9 base editor editing efficacy prediction using the large-scale sgRNA library screen data generated in our lab. The CNN model outperformed a wide variety of machine learning methods. In addition, the model automates the feature identification for sgRNA design in a data-driven manner, empowering the optimization of the editing efficiency prediction. We also used saliency map to facilitate the interpretation of base editor sgRNA design, gaining interpretable results that are consistent with existing biological knowledge.

Our work provides a tool for sgRNA design for large-scale screens and small-scale CRISPR-Cas9 base editing. As base editing is newly developed, there are no large-scale systematically screenings for the sgRNAs of base editor. We synthesized a large library with over 7,000 sgRNAs to introduce pre-mature stop codons into the essential genes. The experiment design enables us to readout the editing efficacy of different sgRNAs in a rapid way. With the valuable data generated from the sgRNA library screening, we carefully designed a deep learning network with parameters and architectures fine-tuning to improve the model’s performance. With the fine-tuning and optimization, our CNN outperformed all the other machine learning methods tested. Discrepancy with restricted assumptions or lack of parameters tuning may cause the less efficient performance for tested machine learning algorithms. In Naïve Bayes Classification (NBC), it assumed the features are independent. However, k-mer features extracted from DNA sequences are not totally independent, which may decrease the power of this algorithm. To further explore how much this assumption affects the prediction, we can compare the performance of NBC using simulated data. One approach is to simulate some features with and without correlation and measure the performance. Due to their characteristic, k-mer features extracted from DNA sequences are always not independent. The main advantage of CNN is its ability to automatically generate and reveal features directly from sequences. Therefore, it is hard to perfectly compare the performance of CNN and NBC. For methods such as KNN and SVM, the appropriate choices of the parameter k and ξ have significant impact on the model performance. For CNN in our study, we have tested different architectures, and tuned the parameters for several layers. It is possible that with carefully parameters tuning, KNN and SVM would have an increase in the model performance.

One restriction for the performance of the current CNN model is the limitation of training data. However, we are participating a rapid increase in the amount of CRISPR base editing given the unique characteristic of this rising tool to convert DNA bases without any other damages to the genome. For this reason, the performance of our CNN model is expected to be enhanced with the availability of more training data. We believe that the expanded training data and the future insights from the deep learning community will lead to enhancement of CRISPR-Cas9 sgRNAs design for base editing.

References

1. Cohen SN, Chang ACY, Boyer HW, Helling RB. Construction of Biologically Functional Bacterial Plasmids In Vitro. Proc Natl Acad Sci. 1973;70: 3240–3244. doi:10.1073/pnas.70.11.3240
2. Hsu PD, Lander ES, Zhang F. Development and applications of CRISPR-Cas9 for genome engineering. Cell. 2014. pp. 1262–1278. doi:10.1016/j.cell.2014.05.010
3. Doudna JA, Charpentier E. The new frontier of genome engineering with CRISPR-Cas9. Science. 2014. doi:10.1126/science.1258096
4. Komor AC, Kim YB, Packer MS, Zuris JA, Liu DR. Programmable editing of a target base in genomic DNA without double-stranded DNA cleavage. Nature. 2016; doi:10.1038/nature17946
5. Balboa D, Weltner J, Eurola S, Trokovic R, Wartiovaara K, Otonkoski T, et al. CasFinder : Flexible algorithm for identifying specific Cas9 targets in genomes. Nucleic Acids Res. 2014; doi:10.1101/005074
6. Heigwer F, Kerr G, Boutros M. E-CRISP: fast CRISPR target site identification. Nat Methods. 2014; doi:10.1038/nmeth.2812
7. Doench JG, Fusi N, Sullender M, Hegde M, Vaimberg EW, Donovan KF, et al. Optimized sgRNA design to maximize activity and minimize off-target effects of CRISPR-Cas9. Nat Biotechnol. 2016; doi:10.1038/nbt.3437
8. Chuai G, Ma H, Yan J, Chen M, Hong N, Xue D, et al. DeepCRISPR: optimized CRISPR guide RNA design by deep learning. Genome Biol. 2018; doi:10.1186/s13059-018-1459-4
9. Zhang Z, Pan Z, Ying Y, Xie Z, Adhikari S, Phillips J, et al. Deep-learning

augmented RNA-seq analysis of transcript splicing. Nat Methods. Nature Publishing Group; 2019;16: 307–310. doi:10.1038/s41592-019-0351-9

10. Lin J, Wong KC. Off-target predictions in CRISPR-Cas9 gene editing using deep learning. Bioinformatics. 2018. doi:10.1093/bioinformatics/bty554

11. Feng PM, Ding H, Chen W, Lin H. Naïve bayes classifier with feature selection to identify phage virion proteins. Comput Math Methods Med. 2013;2013. doi:10.1155/2013/530696

12. Cortes C, Vapnik V. Support-Vector Cortes, C., & Vapnik, V. (1995). Support-Vector Networks. Machine Learning, 20(3), 273–297. http://doi.org/10.1023/A:1022627411411Networks. Mach Learn. 1995;20: 273–297. doi:10.1023/A:1022627411411

13. Ben-Hur A, Weston J. A user’s guide to support vector machines. Methods Mol Biol. 2010;609: 223–239. doi:10.1007/978-1-60327-241-4_13

14. Fukunaga K. The Optimal Distance Measure for Nearest Neighbor Classification. IEEE Trans Inf Theory. 1981;27: 622–627. doi:10.1109/TIT.1981.1056403

15. Cost S, Salzberg S. A Weighted Nearest Algorithm with Symbolic Features. Mach Learn. 1993;10: 57–78. doi:10.1007/BF00993481

16. Zhang Z. Too much covariates in a multivariable model may cause the problem of overfitting. Journal of Thoracic Disease. 2014. pp. E196–E197. doi:10.3978/j.issn.2072-1439.2014.08.33

17. Zhang Z. Introduction to machine learning: K-nearest neighbors. Ann Transl Med. 2016;4. doi:10.21037/atm.2016.03.37