Predicting Human CRISPR Gene-Editing Efficiency with Deep Learning

Max O'Meara
mxomeara@bu.edu

Rehan Samaratunga
rdsam@bu.edu

1 Task

We want to build a CNN that can tell ahead of time whether a tiny "address tag" (RNA string) for genes will work well. In the lab, these tags are short strings that help turn off a chosen part of DNA. Scientists have already tried many tags in cells and measured how strong their effect was. Therefore the goal for the model is to learn these relationships automatically from experimental examples, so that it can make reliable predictions for new RNA sequences it has never seen before. This choice matters because it pushes the model to generalize to genes it hasn't seen before and it must rely on sequence based rules that transfer to other genes rather than relying on properties of specific genes.

In practice, the CNN will learn simple position-specific patterns near the PAM and along the guide (e.g., seed region signals and GC balance) that track with effect strength, producing a calibrated score that ranks candidates across different cell lines and readouts without using gene-level features.

2 Input and Output

User input to our CNN model will be an example CRISPR-Cas9 screen without gene level features. This forces the model to learn universal guide RNA design principles for new genes and RNA sequences it hasn't seen before. The user will input a 23 letter sequence consisting of A/C/G/T, the cell line model, the phenotype, chromosome and the strand.

Concretely, inputs are (seq23, celline, phenotype, chr, strand). seq23 is 23 letters as it includes the guide RNA of 20 characters with the 3 PAM characters which act as a binding and recognition signal for the Cas. celline encodes the experimental context and lets the model adjust baselines across cell types while still relying on sequence rules. phenotype tells the model what outcome is being measured. chr adds minimal genomic context since we exclude coordinates but allows it to contribute to the output without leaking gene specific information. strand indicates the target orientation and if strand is "$-$", we reverse-complement so the CNN always sees the PAM at positions 21–23.

An example input can look like: $GCAGCATCCCAACCAGGTGGAGG$, Jiyoye, viability, 10, $+$. The model will take the input and will output a single number showing the expected strength of the effect using as a Log2-scaled fold change between the start and end of the experiment. An example output can look like: 2.31590732393855947

For each sgRNA, the $\log_2$ fold-change measures the relative change in abundance between the final time point and the initial reference sample. Using the equation,

$$\text{g2fc} = \log_2\left(\frac{N_{\text{final}}}{N_{\text{initial}}}\right),$$

where N_{initial} and N_{final} denote the normalized read counts at time 0 and the endpoint (or untreated vs. treated in positive-selection screens). A negative value indicates sgRNA depletion, while a positive value reflects enrichment.

3 Purpose

The purpose of this project is an attempt to save resources in labs. CRISPR-Cas9 a gene-editing tool that can treat genetic diseases, fight cancer, and combat infectious diseases. When biotech companies are designing new therapies, they're targeting genes they've never screened before. Instead of testing dozens of DNA tags to find one that works, researchers can start with the most promising options our model suggests. That means fewer trial and error experiments, faster progress on basic biology studies, and a smoother path for teams using this tool for disease research, drug studies, or classroom labs.

This reduces library size, sequencing, and hands-on work, freeing time for confirmatory tests and follow-up biology. It also gives a clear, repeatable ranking method that helps small labs and teaching settings. Because the setup is configurable by cell line and phenotype, teams can run quick design–test–update cycles to steadily improve guide selection.

4 Data Used

The data we will be using for our model is provided by GenomeCRISPR which is a database for high-throughput CRISPR/Cas9 screening experiments. The current data set contains data on the performance of approximately 700 000 single guide RNAs (sgRNAs) used in 500 different experiments performed in 421 different human cell lines. The data set is publicly downloadable and can be found at this link: https://genomecrispr.dkfz.de/#!/.

The data contains data on the sgRNA start and end position, the target chromosome, the target strand, the PubMed ID of the screen's original publication, the cell line model used in the screen, the phenotype that was measured in the screen, the sgRNA sequence including PAM, the HUGO gene symbol of the targeted gene, the ENSEMBL gene identifier of the targeted gene, an array of read counts at the 'initial' time point and counts of the treated sample, a value of the sgRNA effect determined by GenomeCRISPR, the Cas variant used in the screen and the type of screen used.

For our model we will not need all of this data since we are training our model to learn generalizable sequence based rules of CRISPR guide efficiency rather than memorizing gene specific effects. This allows the model to learn patterns that can be applied to other genes. To do so we will restrict the data we give our model to just (seq23, cellline, phenotype, chr, strand) which only adds minimal genomic context in the form of including the chr data point. However we are choosing to including the chr data point as it does not teach the model gene identity and excludes genomic coordinates, so it doesn't learn from where a guide is located, only the chromosome level context.

We will divide the GenomeCRISPR data set into 80 percent training, 10 percent validation and 10 percent testing. The data will be split up in a way in which there will be no data leakage as no gene in the training set will appear in the testing set. This reinforces learning of universal genetic patterns as the main idea is to make sure the model is tested on completely new genes that it hasn't seen before, so we can measure how well it generalizes.

5 Performance Evaluation

For model performance, we will use the following metrics to help us evaluate our model: (add equations)

- Spearman Rank Correlation - This will be our primary metric. It measures how well the model's predictions preserve the relative ranking of the true effect scores. This will help the user answer the question of whether or not the model correctly identifies the best-performing guides and the worst-performing (also ranking the best higher than the worst).

$$\rho_s = 1 - \frac{6 \sum_{i=1}^{n} d_i^2}{n(n^2 - 1)} \tag{1}$$

- Pearson Correlation - This will help us measure the linear relationship between the predicted scores and the true effect scores. A high Pearson correlation indicates that the model's output is directly proportional to the true experimental value.

$$r = \frac{n \sum xy - (\sum x)(\sum y)}{\sqrt{(n \sum x^2 - (\sum x)^2)\,(n \sum y^2 - (\sum y)^2)}} \tag{2}$$

  - n = number of data points
  - x and y = variables being compared
  - $\sum$ = summation symbol

- Mean Squared Error (MSE): This will measure the average of the squares of the prediction errors and will give us an outline of the magnitude of a specific error. Also this will penalize larger outlier errors.

$$= \frac{1}{n} \sum_{i=1}^{n} (y_i - \hat{y}_i)^2 \tag{3}$$

We will use all of these metrics but will place a big emphasize on Spearman correlation.

6 Baseline Method

For our baseline model, started with just the sequence only so we used a simple 1D convolutional neural network that takes inputs of shape

$(\text{batch}, 4, 23)$, where the 4 channels are one-hot encodings of A/C/G/T. A single Conv1d$(4 \to 16,\ \text{kernel} = 5,\ \text{padding} = 0)$ scans for 5 metrics, producing $(\text{batch}, 16, 19)$. A ReLU applies nonlinearity, and an adaptive max-pool over the result reduces each channel to its strongest positional response, giving $(\text{batch}, 16, 1)$. We then squeeze to $(\text{batch}, 16)$ and pass through a linear layer $(16 \to 1)$ to produce a scalar prediction per sample. This architecture (353 parameters total) induces position invariance for metric detection while remaining small and quick to train.

7 Baseline Performance

We trained for 3 epochs with Adam (lr $= 10^{-3}$) and MSE loss (batch size $= 256$), the baseline CNN achieves on the validation split: MSE $= 0.6495$, Pearson $r = 0.1399$, Spearman $\rho = 0.1433$, and sign-accuracy $= 0.5220$. On the test split: MSE $= 0.6518$, Pearson $r = 0.1398$, Spearman $\rho = 0.1432$, and sign-accuracy $= 0.5223$. Given the close alignment between validation and test metrics, the results indicate stable generalization.

8 Models

In this section, we describe the progression from our baseline sequence-only model to the full residual multi-scale architecture with structured metadata fusion. Each model incrementally increases representational capacity and biological context-awareness.

Model 1: Baseline Sequence-Only

Our baseline model processes each sgRNA as a 4×23 one-hot encoded sequence. A single Conv1D layer, followed by a ReLU activation and adaptive max pooling, produces a compact feature vector. A linear prediction head outputs the $\log_2$ fold change. This minimalist architecture captures purely sequence-derived signals and serves as a benchmark for evaluating the rest of our models.

Model 2: Sequence + Metadata Embeddings

Building off of the base model, the second model incorporates experiment-level context through learned embeddings for key metadata fields: cell line, phenotype, and chromosome. After extracting sequence features with the same Conv1D module as the baseline, we concatenate them with the metadata embeddings. A linear head then predicts the $\log_2$ fold change from the vector. By including the metadata in the model we found a increase in Pearson Correlation to ≈ 0.19, MSE decrease to ≈ 0.64 and an accuracy increase to ≈ 0.53. By adding the metadata, the model improved slightly in some of it's attributes but not as much as we were hoping.

Model 3: Deep Conv2D Encoder with Multi-Scale Pooling

The third model increases representational capacity by replacing the 1D convolutional framework with a deeper 2D convolutional architecture. The one-hot encoded sequence is reshaped to $(B, 1, 4, 23)$, enabling stacked Conv2D blocks with batch normalization and dropout to identify more spatially structured sequence patterns. To summarize features effectively, we introduced a multi-scale pooling module with two parallel convolutional branches (1×1 and 3×3), each followed by global average pooling and concatenation. Metadata handling is expanded to include a strand embedding, and the prediction head is upgraded to a multi-layer perceptron (MLP) to better capture nonlinear interactions between sequence-derived patterns and metadata. This new model increased it's Pearson Correlation score to ≈ 0.61, Spearman increased to ≈ 0.45, MSE decreased to ≈ 0.42 and testing accuracy jumped to ≈ 0.65. This magnitude of a jump shows that that the sequence encoder is extracting meaningful predictive biological features, while the large MSE drop indicates a substantial reduction in noise.

Model 4: Residual Conv2D + Expanded Multi-Scale Pooling + Interaction Features

The fourth model extends Model 3 with a residual Conv2D block, which improves gradient flow and stabilizes training at greater depth. The multi-scale pooling module is expanded to three branches using 1×1, 3×3, and 5×5 convolutions, combined with both global average and global max pooling. On the metadata side, we add explicit interaction terms between metadata features and use a deeper fusion MLP equipped with a skip connection to preserve low level information. With this model we were able to increase Pearson Correlation to ≈ 0.68, Spearman to ≈ 0.55, MSE reduction to ≈ 0.36 and an increase to $\approx 70\%$ accuracy in our testing data.

9 Conclusion

Our final CNN model substantially improves on the baseline sequence-only model and better captures generalizable rules of CRISPR guide efficiency. The new architecture reduces the validation MSE from 0.6495 to 0.3553 and the test MSE from 0.6518 to 0.3567, nearly halving the average squared error while maintaining tight consistency between validation and test performance. At the same time both the Pearson and Spearman correlations rise from ≈ 0.14 (Spearman and Pearson) at baseline to ≈ 0.68 (Pearson) and ≈ 0.55 (Spearman), indicating that the model captures both the linear trend and rank ordering of the experimental effect sizes more accurately. Finally, the sign-accuracy improves from about ≈ 0.52 to ≈ 0.70, so the model is no longer close to a 50/50 decision whether a guide will either enrich or deplete, and now gets the direction of effect in right in roughly 70% of cases.

Validation - Baseline $\rightarrow$ Final

- MSE: $0.6495 \rightarrow 0.3553$
- Pearson $0.1399 \rightarrow 0.6826$
- Spearman $0.1433 \rightarrow 0.5495$
- Accuracy $0.5220 \rightarrow 0.6991$

Test - Baseline $\rightarrow$ Final

- MSE: $0.6518 \rightarrow 0.3567$
- Pearson $0.1398 \rightarrow 0.6823$
- Spearman $0.1432 \rightarrow 0.5499$
- Accuracy $0.5223 \rightarrow 0.6990$

The close match between validation and test metrics in the final model suggests that these gains are not due to overfitting but show us genuine improvements in how well the model generalizes unseen genes and experimental contexts.

Our experiments showed that the model architecture and feature design matter far more than minor hyperparameter tweaks, as moving from the baseline CNN to our residual multi-scale model with metadata nearly halved the MSE and increased both Spearman and Pearson.

In future work, we could explore more sequence encoders (attention or transformer-based models), more systematic hyperparameter searching, and stronger regularization to further reduce error. We believe that it could also be valuable to add uncertainty estimation, so the model can flag guides where its predictions are unreliable.

Overall, the combination of a deeper residual CNN, multi-scale convolutional pooling, and metadata fusion develops our initial model into a tool that can rank candidate sgRNAs and reduce experimental trial-and-error in CRISPR screen design.

10 Documents

Our models, project board, paper revisions and data can be found at this link: https://github.com/Max-OMeara/CrisprCNN