MEMÒRIA DEL TREBALL DE FI DE GRAU DEL GRAU (ESCI-UPF)

Predicting the template sequence used in CRISPR experiments

AUTOR/A: Marc Escobosa Olmo	**NIA:** 105549
GRAU: Bioinformàtica	
CURS ACADÈMIC: 3r.	
DATA: 15/06/2022	
TUTOR/S: Marc Güell & Mireia Olivella	

Bachelor's Degree in Bioinformatics (UPF-UPC-UB)

Final Grade Project

Predicting the template sequence used in CRISPR experiments

Marc Escobosa Olmo[1]

Scientific director: Marc Güell Cargol

[1]Department of Medicine and Life Sciences, Address Doctor Aiguader, 88 E-08003 Barcelona.

Abstract

Artificial Intelligence and machine learning are becoming more and more popular as years go by, and these technologies are opening lots of opportunities for our society in every way possible. To bring these powerful tools closer to the biological investigation field, we have developed a web application CRISPR-Analytics (CRISPR-A), a tool that analyses and simulates gene editing experiments to help with the evaluation and design of the study. Additionally, my aim was to also make part of the analysis of CRISPR-A automatic, faster and more efficient by building a predictive model that can classify the sequences of reads based on the modifications that happened in each of them. The data used for the construction of the model will be taken from papers that use gene-editing techniques like Prime Editing, Base Editing or Homologous Directed Repair, and from the users of the web application CRISPR-A. The selection of the model is based on performance against a set of toy datasets, the model selected was Random Forest. While the optimization and training of the parameters were performed using the dataset from different studies, the external validation was done using the data given by the users of CRISPR-A. Regarding the parameters used in the model, there have been almost no changes from the initial selection. Finally, the web application was successfully completed, thus allowing users to design and analyse their experiments with the tools developed, while the model was partially developed more time is needed for the external validation and its introduction as a feature for CRISPR-A.

Supplementary information: https://synbio.upf.edu/crispr-a/, https://github.com/olmopolmo/Final_Degree_Project/.

1 Introduction

Clustered regularly interspaced palindromic repeats (CRISPR) is a powerful tool that has allowed us to edit DNA sequences as we desire. The way CRISPR performs these editions is well known and composed of two elements, the Cas (CRISPR Associated) protein —which takes the role of the molecular "scissors"—, and a gRNA —which directs the Cas protein to the desired region of the genome—. Once the system gets to the target location, it performs a DBS (double-strand break) to trigger the cell's natural repair mechanisms. From here, two paths are possible: NHEJ (non-homologous end joining) —which will probably result in not desired modifications such as insertions and deletions— or HDR (homologous directed repair) —which consists of filling the gap based on a short DNA sequence that will act as a template—. This template sequence has the modification we want. Genetic engineering has witnessed a before and after with CRISPR-Cas9, which has opened up an almost limitless number of new opportunities. On top of CRISPR in 2019, David R.Liu developed the Prime Editing technique, Prime editing consists of three elements. First, uses the same "scissors" as CRISPR the Cas protein, with a modification that cuts only in one of the two DNA strands. The second element is a reverse transcriptase, an enzyme that generates a DNA sequence using RNA as a template. The third element is an RNA fragment, called pegRNA (prime editing guide RNA), which has two main functions. On the one hand, It tells the Cas protein where to cut and also, acts as a template for DNA synthesis by reverse transcriptase. Moreover, another technique called Base editing emerged to offer precise single base changes, again this technique uses three components to perform the desired mutation, the Cas protein, a gRNA to tell the Cas where to cut, and a base converting enzyme which performs the desired edit. The introduction of these powerful techniques generated an exponential growth of CRISPR-related experiments leading to an overload of data and therefore a need to evaluate it and visualize the results and analysis in a way that is comprehensible for different scientific fields.

From these needs, tools such as CRISPResso2 (29) —"a tool that allows for an accurate quantification and visualization of the CRISPR results providing an evaluation for the effects on coding sequences, non-coding, and off-target sites" (Pinello et al., 2016)—, CRIS.py (2) —"a python-based program which analyses NGS (Next Generation Sequencing) data

for knock-out and multiple user-specified knock-in modifications from one or many edited samples" (Connelly and Pruett-Miller, 2019)—, CRISPR-GA (3) —"a CRISPR based technology which offers a platform that presents a quality assessment of a genome editing experiment" (Güell, Yang and Church, 2014)—, CRISPRpic (11) "An algorithm created to analyse the sequencing reads for the CRISPR experiments via counting exact-matching and pattern-searching." (Lee, Chang, Cho and Ji, 2020) and crispRvariants (12) "A comprehensive R-based toolkit (CrispRVariants) and an accompanying web tool (CrispRVariantsLite) that resolve and localize individual mutant alleles concerning the endonuclease cut site." (Lindsay et al., 2016) emerged over the years to fulfil these needs. However, these tools show some limitations. As an example, these tools do not include interactive plots, template sequence discovery, simulations to help in design, benchmarking, reference identification, clustering or noise subtraction. From these necessities appears CRISPR-Analytics —an updated and improved version of CRISPR-GA (3)—.

1.1 CRISPR-ANALYTICS

CRISPR Analytics (CRISPR-A) emerges as an evolution of CRISPR-GA (3), the first NGS-based method for gene-editing assessment. CRISPR-A is a very versatile and user-friendly tool that offers a wide range of customizable parameters that can make the analysis of the investigator's data easier and more precise. CRISPR-A offers the option to evaluate the results from a variety of single-cut experiment methods, including Base Editing (BE), Prime Editing (PE), and Homologous Directed Repair (HDR), among others. Moreover, if the reference sequence is not provided the application will search for an appropriate new reference to be used. CRISPR-A has been my main job, since July 1st of 2021, in the synthetic biology laboratory at UPF. I contributed to its front-end development and visualization of results. From here I was able to start developing my predictive model to then become an additional feature that the application could offer. Currently, the web of CRISPR-A is available for the public with the warning of being a beta version (https://synbio.upf.edu/crispr-a/). To this day we are constantly receiving feedback from the users about issues and suggestions, to improve the client interface, the algorithm efficiency and the visualization of the results.

1.2 Model building

As said, while the reference can be obtained without the user providing it, we cannot state the same for the template sequence. It would be helpful to perform a prediction based on machine learning algorithms to be able to analyse without the user providing the template sequence. This predictive model would make the application more flexible; moreover, it might help detect modifications that initially would have been ignored or wrongly classified by the base algorithm. On top of that, the analysis of the reads would also be faster, given that a predefined model would be in charge of the classification.

The current CRISPR-A algorithm arranges reads into 11 groups: deletions, deletions inframe, deletions outframe, insertions, insertions inframe, insertions outframe, delins, indels, template-based modification, substitutions, and wild types, along with their respective percentages against the total amount of reads and also showing the microhomology patterns that have led to deletions.

The bioinformatics team of translational synthetic biology and I discussed the potential to be able to predict these groups based on the protospacer, reference, and the reads that the user provides —which in theory is possible, given that some clear patterns show us which reads have suffered the desired modification and which do not—. The objective of this project would be to participate in the development of CRISPR-A and create a machine learning algorithm that can predict template modifications having only the reference —given by the user or the algorithm—, the reads and the gRNA. The model should be able to predict the template modification regardless of which approach has been used to make the template modification —Homologous Directed Repair (HDR), Prime Editing (PE) or Base Editing (BE)—. The first step will be to select the optimal model for the problem hence we can later optimise the chosen model and its parameters.

Machine learning is a branch of artificial intelligence based on self-learning algorithms. These self-learning algorithms are really powerful tools that can be used to assess biological problems, especially those related to prediction and discovery, such as genomic features. We can classify the algorithms that machine learning offers into 6 groups: Text analytics, Multiclass Classification, Regression, Two-Class classification, Image classification, and Clustering, all of which can be easily applied to the biological world. For the aim of my project, both clustering and multiclass classifications were initially considered.

1.2.1 Decision Trees & Random Forest

Decision Trees are supervised algorithms that classify data by asking questions —usually, yes-no questions—. Each question of the Tree is a node, and from this node, two children appear, a no-child and a yes-child. From each child, the Decision Tree asks a new question, generating more branches until we reach the leaves, that would correspond to each of the possible classifications that can occur. Due to this process, the algorithm is very dependent on the selection of the questions. Moreover, a small change in the input data may cause a big change in the structure of the tree. The good part about Decision Trees is that they are easily interpretable. The data is also classified quickly once computed, and both categorical and numerical data can be used in the training process.

One big issue with decision trees is overfitting. To overcome this problem, the usage of Random Forests can provide robustness. In the Random Forest, an algorithm creates several trees that have different questions (nodes) and are uncorrelated between them. The best-performing tree is selected from all of the selections.

1.2.2 Neural Networks

The structure of Neural Networks is based on the neural connections of the human brain. This algorithm is used for classification, although it only allows numerical data to be introduced. The structure of the Neural Network is divided into 3 blocks: the input layer, one or more hidden layers, and the output layer. Each of these layers is made of several nodes that are connected to the following layer.

The problem with Neural Networks is that they only accept numerical data. Therefore, there are clear limitations when choosing the parameters that describe my data set.

1.2.3 Principal Component Analysis (PCA)

This algorithm is used to reduce the dimensionality of a given data set while trying to keep the maximum amount of variation. The dimensionality of the data gets reduced to the principal components (PC) which are indeed linear combinations of the original variables. Taking some of the principal components can explain most of the variation in our data, allowing us to then plot the data based on these PCs and try to identify differences and similarities to group them. The main problem with PCA is in its simplicity, as quantifying the variation in one direction ignores other possible directions.

1.2.4 t-Distributed Stochastic Neighbor Embedding (tSNE)

tSNE is an unsupervised algorithm designed to visualise high-dimensional data. And its usage after the dimensionality reduction of PCA is highly recommended, as it seems to increase the performance of the tSNE. In the case of tSNE, the hyperparameters of perplexity, the number of steps and epsilon will add an extra layer of complexity to the model. The problem, however, is that tSNE is used for unsupervised data-frames and it only accepts numerical data.
The problems for PCA and tSNE are suited for unsupervised data and only accept numerical data. Nevertheless, I could see if the classifications of the reads could be done using these clustering algorithms if the appropriate parameters are given.

1.2.5 Uniform manifold approximation and projection (UMAP)

Once more, this method is very useful for high dimensionality. UMAP tries to group in clusters based on the distance of "neighbourhoods" in the dataset, aiming to reduce the dimensionality as much as possible while preserving the "neighbourhoods". The problem with it is that the distances between points are harder to interpret. However, UMAP shows some advantages. The most important one is memory usage. It requires less memory than tSNE and preserves better the overall structure of the data when clustering.

All these approaches seem likely to lead to good results, given that they are the most suitable to assess the classification and analysis of reads and their modifications. I plan to go over all of these techniques to find those that truly work and pick the optimal one from those. Once we obtain the optimal tool the next and final step will be to build the dataset from which I will optimise the parameters and build the optimal predictive model.

2 Methods

2.1 CRISPR-A

CRISPR-A web link: **https://synbio.upf.edu/crispr-a/**

2.1.1 Pipeline

In order to run, the CRISPR-A pipeline needs the following parameters: One or two fastq files —Depending if the sequencing was done as Paired-end or Single-End— containing the reads which can either be simulated or uploaded, a protospacer sequence (optional), the template sequence (optional), and a reference sequence (optional). If the reads uploaded are paired-end, we use PEAR (19) to merge them, then using FastQC (18) and Cuatadapt (20) we detect and cut the adapters. After performing an initial quality control using fastq_quality_trimmer from fastx-toolkit (28), an adapted version of the extract_umis.py script from the pipeline_umi_amplicon pipeline (provided by NanoporeTech https://github.com/nanoporetech/pipeline-umi-amplicon) is used to obtain UMI sequences from the reads. Vsearch (21) is then used to cluster UMI sequences. UMIs are polished with minimap2 (22) and racon (23) and consensus sequences are obtained with minialign and medaka. The reads are aligned using BWA-MEM (25), instead of the amplicon reference sequence. Next, we can find the reference if it isn't provided by using samtools (24), bedtools (27) and custom scripts. Afterwards, the amplicon sequence is oriented in the same direction as the gRNA, after which the reads are aligned against the reference using minimap2 (22). Custom scripts can then be used to extract some comparable data. Through the scripts, variant calling, error correction, and noise subtraction are performed. The variant calling process is based on the cigar from the alignment between the reads and the reference using some scripts. From these results we generate some tables and plots.

2.1.2 Visualization

Based on the results, we display all plots and tables on a results page, subdividing the page into different categories. Two tables (Fig.1) summarise the core parameters, such as the number of editions per class (Template-based, Indels, Delins, Insertions, Insertions Inframes, Insertions Outframes, Deletions, Deletions Inframes, and Deletions Outframes), the number of Aligned Reads, the number of Clustered Reads, the number of Adapters, the total number of reads, the number of reads that were merged, and the number of reads that went through the quality filter.

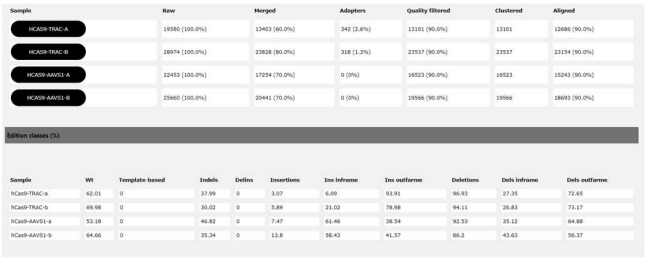

Fig. 1. Summary table of the samples processed.

Additionally, each sample has a button that displays the IGV interface with the reads, provided by the user, aligned against the amplicon sequence. Thanks to the IGV the user has access to a more detailed description of their reads, moreover it allows the user to have a first visualisation of the quality of its alignment and get a picture of their results. In addition, the reads are sorted by insertion size in descending order.

Fig. 2. IGV interface of sample HCAS9-TRAC-A.

Then a heatmap with Jensen distances of indel positions and length showing how related are the samples given by the user.

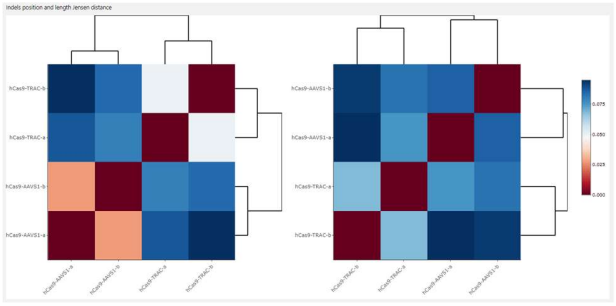

Fig. 3. Heatmap comparing Jensen distances between samples.

Moreover, for each sample, we show three pie charts showing a summary of the pre-processing, the percentages of the reads that went through the quality filters, and the editions that we observed.

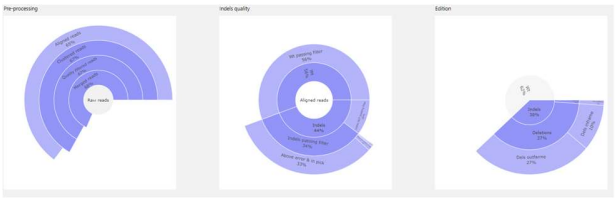

Fig. 4. Pie charts showing the Edition summary of sample HCAS9-TRAC-B.

Following these plots, we dig deeper into the modifications separately. Our interactive plots allow users to take a closer look at their analysis: we show them the frequencies of the positions where deletions/insertions started, and next to that, a plot of the sizes of the deletions/insertions. In addition, the user can select which microhomology patterns are shown in the plot, and zoom on the desired regions.

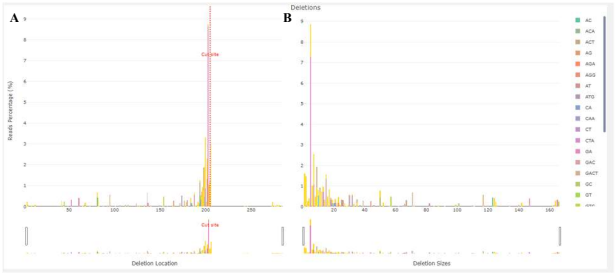

Fig. 5. A, Barplot showing the percentage for the starting position for deletions per position in the sequence. **B**, Barplot representation of the percentage of deletions sizes.

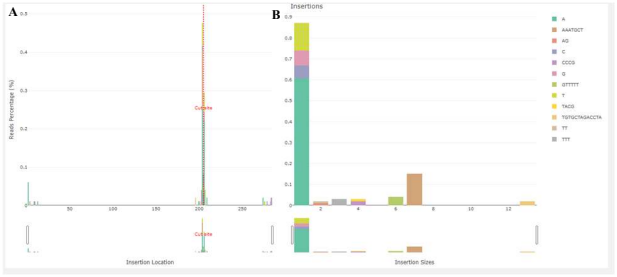

Fig. 6. A, Barplot showing the percentage for the starting position for insertions per position in the sequence. **B**, Barplot representation of the percentage of insertion sizes.

Using the previous plot as a starting point, we created an accumulative plot that represents all the modifications frequency by position. Here the user can also select the region of interest, and the graph will calculate again the percentage of modifications that represent the new region, additionally, it will display only the modifications with start or end positions within the range selected.

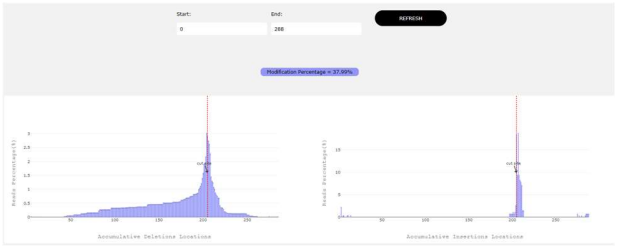

Fig. 7. Accumulative plots, count from the start position to the end of the modification, making the y axis represent the number of modifications per position divided by the total amount of Deletions or Insertions.

2.2 Model Selection

2.2.1 Background

The first step is to look at the current pipeline of the CRISPR-A application to see which parameters are calculated and which files are available before the classification algorithms run, the available parameters are the following: cut site (numeric), the reference sequence, and the bam file created from the alignment of the reference against the reads.

The Bam file is the central pillar of the data frame, it provides three columns: the CIGAR, position where the alignment begins, sequence, and the name/id of the sequence. —To simplify the columns will be called cig, pos, seq, and qname respectively. —

2.2.2 Planning

This is all the a-priori information that is available to me, from here I will take 11 toy data sets and obtain all the data mentioned earlier.

Now comes the thinking part, from these variables how do I get significant parameters to properly train the models? Discussing with the team we reached a list of properties that could be useful:

- Size: Size of the modification. (numeric)
- Modification: What modification is present in the CIGAR. (categorical)
- Distance to cut site: How far is the modification from the cut site. (numeric)
- GC content: percentage of GC content in the sequence. (numeric)
- Frequency: How many times do the same modification and sequence appear. (numeric)
- Wild type similarity: A score of the alignment of each sequence against the wild type. (numeric)
- Group: The variable that has the actual modification of the sequence, will be used to test the accuracy of the models. (categorical)
- Base Modification: In the case of substitution which was the change. (categorical)

The idea is to get and merge all the information and parameters mentioned so far, from each toy data set, into a data frame using RStudio.

2.2.3 Data frame creation

I will list how I should obtain the different parameters listed in the previous point.

- Size: For deletions and insertions is straightforward by taking the number of the CIGAR that is followed by a "D" (for deletions) and "I" (for insertions). In the case of wild types I decided to assign size 0, and for substitutions, I take the number of consecutive substitutions that are nearest to the cut site. — The problem with substitutions is that it is very likely that in the extremes of the sequences there is some noise in the form of unwanted substitutions. —
- Modification: Again, for the deletions and insertions there is no problem if the character is a "D" the modification is set to "del" if it is an "I" the modification is set to "ins". If there is any substitution in a range of 20 nucleotides upstream and downstream then I classified it as a substitution "sub", if not the sequence is classified as wild type "wt".
- Distance to the cut site: I obtain the start of the modification by looking at the number of "M" after "I" or "D" in the cigar for insertions and deletions, for substitutions I use the first position where the substitution happened, and in the case of wild types I set this variable to length of the sequence. Then by subtracting the start of the modification to the cut site I can easily obtain the distance.
- GC content: percentage of GC in the sequence.
- Frequency: Using the function in RStudio Freq() I can get this parameter.
- Wild type similarity: Doing a pairwise alignment of each sequence against the wild type this value is easily obtained.
- Group: This is obtained by running the whole algorithm of CRISPR-A on the toy data sets.
- Base Modifications: If there is no substitution this parameter has the value "-". It is obtained by simply looking at the change that happened in the sequence.

Once all these parameters are computed for the toy data sets, I will save them in a CSV file just to save time.

2.2.4 Pre-processing

Before any testing, the data frame built from the parameters explained earlier will be treated. Cleaning the data by eliminating every row that contains NAs and normalising the numerical values.

Creating a train and test data frame containing 70% and 30% of the toy data frame respectively. —Given that this is a toy data frame I will not use any external validation for the models, nonetheless, when I reach the point of using the real data some extra considerations in the pre-processing will be taken into account. —

2.2.5 Model Comparison

To compare the models, I will calculate the accuracy of each. The ones with higher accuracy will be used later with real data, the idea of this toy data is only to see which models are useful. The optimization of which model and which parameters are optimal will be performed with real data sets.

2.3 Model Creation

2.3.1 Data research

Once the model is selected, the priority goes to finding a real and big dataset with all the elements and characteristics that are known to be impactful to the model. The dataset must include prime editing (PE), base editing (BE), and homology-directed repair (HDR) data. Moreover, in the case of PE and HDR, subsets of deletions, insertions, and substitutions are needed. To fulfil this need I started to read several articles and look forward to finding a paper that has done CRISPR modifications —with either PE or HDR— so I could get the data. This turned out to be way harder than initially expected, most of the papers didn't have the data that they used for the experiments in a way that can be used for a bioinformatician. Two papers https://www.nature.com/articles/s41586-019-1711-4, and https://www.nature.com/articles/s41587-021-01133-w had an insane amount of data from both PE and HDR, enough to build my model.

2.3.1 Data

The data set built for the train and test of the model contains a total of 580 samples; from HDR a total of 187 samples, 138 substitutions, 25 insertions and 24 deletions, for PE a total of 433 samples, 318 substitutions, 42 insertions and 42 deletions. The real dataset will be treated to have the same parameters as the toy datasets with a small but very significant change. The previous "Modif" parameter was used as a variable in the formula for building the Random Forest, however, turned out to be much more efficient to previously separate the data frame based on the "Modif" column and do 3 different Random Forest one for deletions, one for Insertions and one for substitutions.

2.3.1 Preprocessing

To perform appropriate testing, I will first pre-process the data, in a similar approach as the one taken for the toy data, I will get rid of the rows that contain NA and normalise all the numerical values. From here the plan is to distribute the dataset in the following manner 60% for training 30% for testing and the remaining 10% that will remain untreated for the external validation.

2.3.1 Testing

To test the model, we will apply the same approach as what we did to select the model. However, this time I will iterate with the same model making changes or additions to the dataset parameters to obtain more accurate results.

GitHub link: https://github.com/olmopolmo/Final_Degree_Project/.

3 Results and Discussion

3.1 Model Selection

3.1.1 PCA

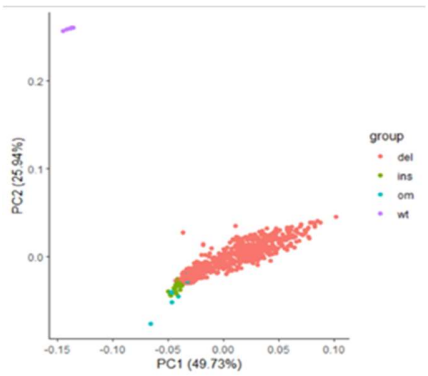

Fig. 8. PCA results from the toy dataset. Scatter plot where each dot represents the classification of each group of sequences. Each colour corresponds to a different classification: red for deletions "del", green for insertions "ins", cyan for template-based modification "om" and purple for wild types "wt".

When running PCA over the toy dataset the results were surprisingly bad, it seemed that the wild type was differentiable. Nonetheless, I cannot state the same for the other modifications, these cannot be clustered using PCA. Therefore, I decide to discard any further approach with PCA.

3.1.2 tSNE

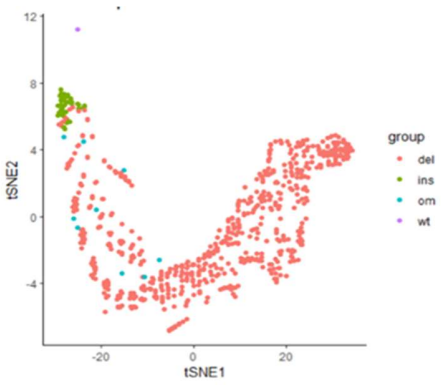

Fig. 8. tSNE results from toy dataset. Scatter plot where each dot represents the classification of each group of sequences. Each colour corresponds to a different classification: red for deletions "del", green for insertions "ins", cyan for template-based modification "om" and purple for wild types "wt".

tSNE did not show any better results than PCA. The wild type was separated from the rest. The other modifications are not clustered using this approach. In the case of tSNE, I played with the parameters of perplexity and number of steps but it d0069d not change the results. I discard the usage of tSNE for this project

3.1.3 UMAP

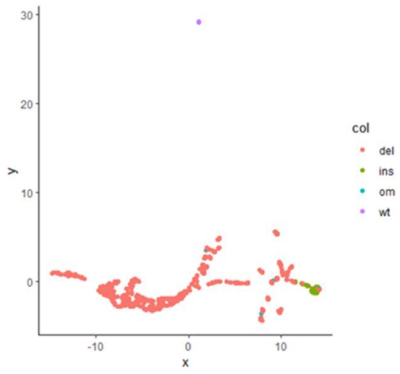

Fig. 8. UMAP results from toy dataset. Scatter plot where each dot represents the classification of each group of sequences. Each colour corresponds to a different classification: red for deletions "del", green for insertions "ins", cyan for template-based modification "om" and purple for wild types "wt".

Using UMAP seems the same case as tSNE and PCA, only the wild types are distinguishable.

Due to the results obtained with all the clustering methods, I can confidently say that for the classification that my project tries to approach, the clustering methods are not good approaches.

3.1.4 Artificial Neural Network

Table 1. Results of Neural Network algorithm

Modification	Del	Ins	Om	Wt
Del	215	0	0	0
Ins	1	13	0	0
Om	1	0	2	0
Wt	0	0	0	4

The string "om" refers to the objective modification that is the same as template modification

Artificial Neural Networks seem to go in a better direction. Even though these results are not good, maybe by giving a bigger dataset and some extra parameters this model could work.

3.1.4 Random Forest & Decision Trees

Table 2. Results of Random Forest algorithm

Modification	Del	Ins	Om	Wt
Del	215	0	0	0
Ins	0	14	0	0
Om	0	0	3	0
Wt	0	0	0	4

The string "om" refers to the objective modification that is the same as template modification

We saw that both Decision Trees and Random Forest gave the best results, with 100% accuracy over the toy samples (obviously these results are a product of overfitting, but these tests are not to build the model but to see which is better). Knowing that Random Forest offers' a more robust algorithm against the overfitting I decided that would become the main algorithm to build the model.

3.2 Final Model Results

3.2.1 Random Forest – Deletions & Insertions

Table 3. Results of Random Forest algorithm against Deletions

Modification	Del	Om
Del	75	0
Om	0	11

The string "om" refers to the objective modification that is the same as template modification

Table 4. Results of Random Forest algorithm against Insertions

Modification	Ins	Om
Ins	53	0
Om	0	12

The string "om" refers to the objective modification that is the same as template modification

We can observe that the accuracy of the Random Forest against the deletions and insertions is 100% given that the data used had a wide variety of both deletions and Insertions I am quite comfortable with the generated model.

3.2.2 Random Forest – Substitutions

Table 5. Results of Random Forest algorithm against Substitutions

Modification	Subs	Om
Subs	159	20
Om	18	86

The string "om" refers to the objective modification that is the same as template modification

Even though the results aren't bad, I am not comfortable with the results obtained. Parameters such as "cut_site_dist" and "size" turned out to be less significant for substitutions than for deletions and insertions, this might be because substitutions sizes were in the range of 1 to 4 bases, given the short sizes, it is more likely for random small mutations to confuse the model. in this data frame which can lead to random small mutations to confuse the model. Still, 86% of accuracy while not acceptable it's close to the desired percentage, I believe that with the addition of other parameters and the removal of the less significant I can get to the desired accuracy of at least 90%.

4 Conclusions

CRISPR-Analytics was created to retrieve all kinds of events in NGS, to add flexibility to cover all genome tools diversity, to add simulations to aid in design and analysis, to add an interactive results interface that helps to receive a deeper understanding of your experiment, to detect errors, and to reduce noise. Almost all of these properties have been successfully included in the web tool, and the properties left to be added are in the final stages of development. The purpose of the predictive algorithm for target sequences in CRISPR experiments was to add an extra layer of flexibility to the experiment, allowing the analysis of template-based modifications without the user providing the template itself. For deletions and insertions, 100% accuracy was achieved, and for substitutions, 86% accuracy was achieved. To increase the accuracy of the substitutions model, I will modify the data frame and collect some more data to cover more types of substitutions.

Acknowledgements

This project and research could not have been possible without the support of my scientific director and supervisor, Marc Güell Cargol and Marta

Sanvicente, and my work colleague Albert Garcia. Their knowledge and their cheering personality have been my inspiration from beginning to end, giving me advice when I needed it the most. Also special thanks to the Bioinformatician Júlia Mir who was always available if I needed help with any bioinformatics stuff.

I am also grateful for the warm welcome by the members of the synbio lab for having such a friendly and hard-working environment.

.

References

1. Anzalone, A., Randolph, P., Davis, J., Sousa, A., Koblan, L., Levy, J., Chen, P., Wilson, C., Newby, G., Raguram, A. and Liu, D., 2019. Search-and-replace genome editing without double-strand breaks or donor DNA. *Nature*, 576(7785), pp.149-157.
2. Connelly, J. and Pruett-Miller, S., 2019. CRIS.py: A Versatile and High-throughput Analysis Program for CRISPR-based Genome Editing. *Scientific Reports*, 9(1).
3. Guell, M., Yang, L. and Church, G., 2014. Genome editing assessment using CRISPR Genome Analyzer (CRISPR-GA). *Bioinformatics*, 30(20), pp.2968-2970.
4. Pinello, L., Canver, M., Hoban, M., Orkin, S., Kohn, D., Bauer, D. and Yuan, G., 2016. Analyzing CRISPR genome-editing experiments with CRISPResso. *Nature Biotechnology*, 34(7), pp.695-697.
5. Ao, C., Yu, L. and Zou, Q., 2020. Prediction of bio-sequence modifications and the associations with diseases. *Briefings in Functional Genomics*, 20(1), pp.1-18.
6. Anzalone, A., Randolph, P., Davis, J., Sousa, A., Koblan, L., Levy, J., Chen, P., Wilson, C., Newby, G., Raguram, A. and Liu, D., 2019. Search-and-replace genome editing without double-strand breaks or donor DNA. *Nature*, 576(7785), pp.149-157.
7. Diaz-Papkovich, A., Anderson-Trocmé, L. and Gravel, S., 2020. A review of UMAP in population genetics. *Journal of Human Genetics*, 66(1), pp.85-91.
8. Hsu, J., Grünewald, J., Szalay, R., Shih, J., Anzalone, A., Lam, K., Shen, M., Petri, K., Liu, D., Joung, J. and Pinello, L., 2021. PrimeDesign software for rapid and simplified design of prime editing guide RNAs. *Nature Communications*, 12(1).
9. Kingsford, C. and Salzberg, S., 2008. What are decision trees?. *Nature Biotechnology*, 26(9), pp.1011-1013.
10. Ringnér, M., 2008. What is principal component analysis?. *Nature Biotechnology*, 26(3), pp.303-304.
11. Lee, H., Chang, H., Cho, S. and Ji, H., 2020. CRISPRpic: fast and precise analysis for CRISPR-induced mutations via prefixed index counting. *NAR Genomics and Bioinformatics*, 2(2).
12. Lindsay, H., Burger, A., Biyong, B., Felker, A., Hess, C., Zaugg, J., Chiavacci, E., Anders, C., Jinek, M., Mosimann, C. and Robinson, M., 2016. CrispRVariants charts the mutation spectrum of genome engineering experiments. *Nature Biotechnology*, 34(7), pp.701-702.
13. Synbio.upf.edu. 2022. *CRISPR-A*. [online] Available at: <https://synbio.upf.edu/crispr-a/> [Accessed 21 March 2022].
14. Anzalone, A. v., Gao, X. D., Podracky, C. J., Nelson, A. T., Koblan, L. W., Raguram, A., Levy, J. M., Mercer, J. A. M., & Liu, D. R. (2021). Programmable deletion, replacement, integration and inversion of large DNA sequences with twin prime editing. *Nature Biotechnology 2021 40:5*, *40*(5), 731–740. https://doi.org/10.1038/s41587-021-01133-w
15. Schubert, M. S., Thommandru, B., Woodley, J., Turk, R., Yan, S., Kurgan, G., McNeill, M. S., & Rettig, G. R. (2021). Optimized design parameters for CRISPR Cas9 and Cas12a homology-directed repair. *Scientific Reports 2021 11:1*, *11*(1), 1–15. https://doi.org/10.1038/s41598-021-98965-y
16. Abby, S. S., Néron, B., Ménager, H., Touchon, M., & Rocha, E. P. C. (2014). MacSyFinder: A Program to Mine Genomes for Molecular Systems with an Application to CRISPR-Cas Systems. *PLOS ONE*, *9*(10), e110726. https://doi.org/10.1371/JOURNAL.PONE.0110726
17. Park, J., Yoon, J., Kwon, D., Han, M. J., Choi, S., Park, S., Lee, J., Lee, K., Lee, J., Lee, S., Kang, K. S., & Choe, S. (2021). Enhanced genome editing efficiency of CRISPR PLUS: Cas9 chimeric fusion proteins. *Scientific Reports 2021 11:1*, *11*(1), 1–9. https://doi.org/10.1038/s41598-021-95406-8
18.
19. S. Andrews, Others, FastQC: a quality control tool for high throughput sequence data (2010).
20. J. Zhang, K. Kobert, T. Flouri, A. Stamatakis, PEAR: a fast and accurate Illumina Paired-End reAd mergeR. *Bioinformatics*. **30**, 614–620 (2014).
21. M. Martin, Cutadapt removes adapter sequences from high-throughput sequencing reads. *EMBnet.journal*. **17**, 10–12 (2011).
22. T. Rognes, T. Flouri, B. Nichols, C. Quince, F. Mahé, VSEARCH: a versatile open source tool for metagenomics. *PeerJ*. **4**, e2584 (2016).
23. H. Li, Minimap2: pairwise alignment for nucleotide sequences. *Bioinformatics*. **34**, 3094–3100 (2018).
24. R. Vaser, I. Sović, N. Nagarajan, M. Šikić, Fast and accurate de novo genome assembly from long uncorrected reads. *Genome Res.* **27**, 737–746 (2017).
25. P. Danecek, J. K. Bonfield, J. Liddle, J. Marshall, V. Ohan, M. O. Pollard, A. Whitwham, T. Keane, S. A. McCarthy, R. M. Davies, H. Li, Twelve years of SAMtools and BCFtools. *GigaScience*. **10** (2021), , doi:10.1093/gigascience/giab008.
26. H. Li, R. Durbin, Fast and accurate short read alignment with Burrows-Wheeler transform. *Bioinformatics*. **25**, 1754–1760 (2009).
27. A. R. Quinlan, I. M. Hall, BEDTools: a flexible suite of utilities for comparing genomic features. *Bioinformatics*. **26**, 841–842 (2010).
28. A. Gordon, G. J. Hannon, Others, Fastx-toolkit. *FASTQ/A short-reads preprocessing tools (unpublished) http://hannonlab. cshl. edu/fastx_toolkit*. **5** (2010).bedtools
29. Clement, K., Rees, H., Canver, M.C. *et al.* CRISPResso2 provides accurate and rapid genome editing sequence analysis. *Nat Biotechnol* 37, 224–226 (2019).