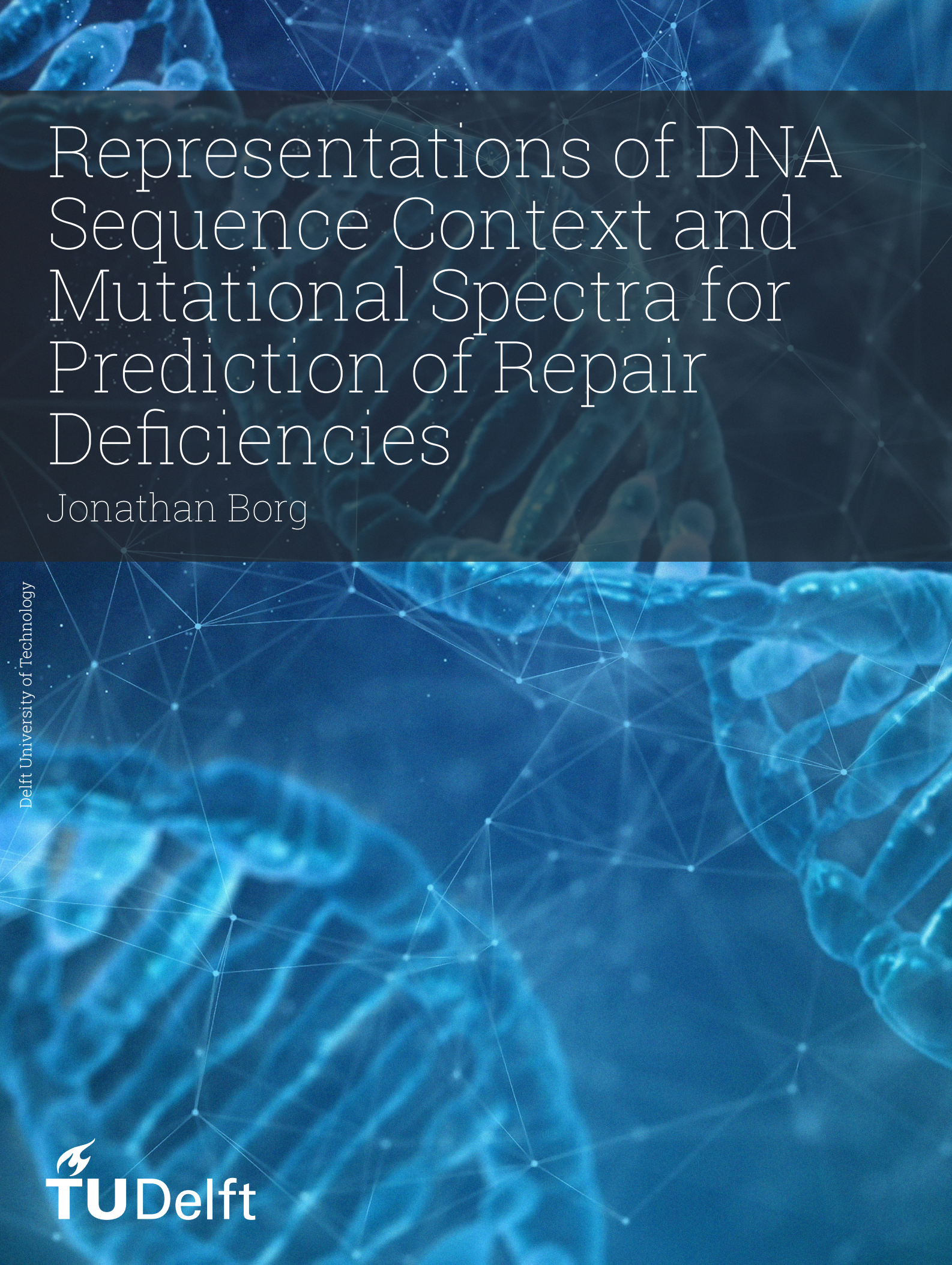
Representations of DNA Sequence Context and Mutational Spectra for Prediction of Repair Deficiencies

Jonathan Borg

Delft University of Technology



Representations of DNA Sequence Context and Mutational Spectra for Prediction of Repair Deficiencies

by

Jonathan Borg

to obtain the degree of Master of Science
at the Delft University of Technology,
to be defended publicly on Thursday November 2, 2024 at 1:00 PM.

Student number:	5613841	
Masters programme:	Computer Science, Bioinformatics specialization	
Faculty:	Electrical Engineering, Mathematics and Computer Science	
Project duration:	November 14, 2023 – November 2, 2024	
Thesis committee:	Dr. Joana S. de Pinho Gonçalves,	TU Delft, supervisor
	Dr. Jorge A. Martinez Castaneda,	TU Delft
	MSc. Colm F. Seale	TU Delft, daily supervisor

An electronic version of this thesis is available at `http://repository.tudelft.nl/`.

TU Delft

Preface

This thesis represents the culmination of a twelve-month research project in bioinformatics and computer science. After the global pandemic, I started asking myself how can a computer scientist aid with some of the biological problems we face. After some soul-searching, I left the software engineering scene and continued my computer science studies, specialising in bioinformatics. Now that the end is near, I am excited to see what the future holds. The journey leading up to this point came with its ups and downs. But luckily, along the path, I met some fantastic people who left an everlasting impact on my life.

Throughout this work, I have had the privilege of receiving advice, inspiration, and support from numerous individuals. Firstly, I would like to thank my supervisor, Joana Gonçalves, for her patience, guidance and support. Her unwavering commitment to academic excellence and invaluable insights have shaped this research. Secondly, I would like to thank my daily supervisor, Colm Seale. His guidance and mentorship have refined my understanding of the subject matter and encouraged me to explore new avenues of inquiry. Thirdly, I would like to thank my committee member, Jorge Martinez Castaneda, for his time and interest in the research. Additionally, I would like to express a heartfelt thank you to my friends Varnika, Frans, Johannes and especially Nicholas for always being there and making this journey an amazing one. Lastly, I would like to thank my family and my significant other, Stefania. Your unwavering encouragement and belief in my abilities have been a constant source of motivation. I hope this work may serve as a source of inspiration and insight for future researchers and scholars in this field.

Jonathan Borg
Delft, November 2023

Representations of DNA Sequence Context and Mutational Spectra for Prediction of Repair Deficiencies

Jonathan Borg,[1,*] Colm Seale[1] and Joana de Pinho Gonçalves[1]

[1]Pattern Recognition and Bioinformatics, Intelligent Systems Dept., EEMCS Faculty, Delft University of Technology, The Netherlands

*Corresponding author. j.borg@student.tudelft.nl

Abstract

Double-strand break (DSB) repair is a critical cellular process which repairs breaks in both strands of the DNA double helix. Different repair mechanisms are tasked with repairing such breaks. Predicting deficiencies in repair mechanisms has been widely used for therapeutic purposes, such as targeting cancer cells that have specific DNA repair deficiencies. DSB repair, however, is not error-free, resulting in mutations. These mutations are also influenced by the DNA sequence surrounding the break site. To the best of our knowledge, sequence representations have not been considered when predicting DNA repair deficiencies. We hypothesise that higher-order information can be extracted from sequence representations. In this study, we research the problem of predicting Non-Homologous End Joining (NHEJ) repair deficiencies. Initially, we evaluate how accurately we can predict NHEJ repair deficiency using only the mutational outcome frequencies (mutational spectra). Afterwards, we examine how combining mutational spectra with representations of the sequence surrounding the break site can improve the prediction of NHEJ repair deficiency. We demonstrate that adding DNABERT sequence representations to mutational spectra features significantly improves prediction accuracy from 94.44% to 96.12%. We also show that even simple sequence representations, such as 1-mer frequencies, can lead to significant improvements. Our findings highlight the importance of including sequence representations with mutational spectra in repair deficiency prediction.

Key words: DNA double-strand break, repair pathway deficiency, mutational spectra, DNA sequence representation

1. Introduction

DNA double-strand break (DSB) repair is a critical process in molecular biology. It ensures the stability and fidelity of genetic material by repairing breaks in both DNA double helix strands [1]. If untreated, DSBs can lead to genomic instability, mutations, and, in some cases, cell death [2]. In mammalian cells, three primary repair pathways are believed to address DSBs in DNA [3, 4]: Homologous Recombination (HR), Non-Homologous End Joining (NHEJ) and Microhomology-Mediated End Joining (MMEJ). However, repair pathways are not error-free, resulting in alterations (mutations) to the DNA sequence. An accumulation of these mutations has been linked to ageing and diseases like cancer [3, 4, 5].

While the HR pathway is less prone to mutations, the NHEJ and MMEJ pathways have been found to generate more errors in their repair products [4, 5]. The two major categories of mutations that cause many human genetic variants associated with the disease are insertions and deletions [6]. Insertions introduce nucleotides to the sequence, whereas deletions remove nucleotides, as depicted in Figure 1. When studying these repair pathways, scientists found that different pathways tend to be prone to certain mutations more than others. For example, NHEJ is prone to insert/delete shorter sequences than other pathways [3, 4].

Understanding the relationship between DNA repair pathways and mutations can allow us to understand the biological processes better. This allows us to apply these findings to different studies, such as targeted treatments.

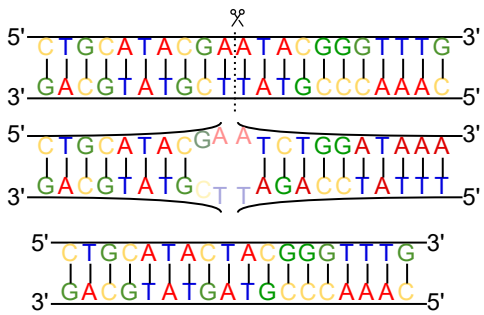


Fig. 1: A double-strand break (DSB) in DNA and its repair product, with a deletion of three nucleotides. Top illustration (DNA sequence): DNA consists of the forward ($5' - 3'$) and reverse ($3' - 5'$) strands. The DSB occurs on the forward strand between two "A" nucleotides, as shown, and the reverse strand between two "T" nucleotides. Middle illustration (DSB): The DSB causes a break on both strands, resulting in the loss of bond between the nucleotides on either side of the break. During the repair process, the repair pathway attempts to amend the break and deletes the nucleotide "A" on the right and the nucleotide "GA" on the left of the cut site. Bottom illustration (repair product): Once both strand ends are reannealed, the repair product no longer contains the sequence "GAA" on the forward strand or "CTT" on the reverse strand.

Different researchers have looked into repair pathway deficiency within different contexts and applications. For example, Davies et al. [7] created a logistic regression model to predict when BRCA1/BRCA2 genes are deficient. They achieved this using mutational signatures, that is, characteristic combinations of mutation types derived from specific processes. Such mutations were collected using whole genome sequencing (WGS), which is the process of determining the entirety of the DNA sequence of an organism's genome.

Alternatively, Shen et al. [6] used a deep neural network (DNN) to predict the frequencies of insertion and deletion products from double-stranded break repair. Instead of using mutational signatures and WGS, they observe the mutation spectra using CRISPR-Cas9 [8]. The mutational spectra are the frequency distributions of mutational outcomes generated under specific conditions. They used CRISPR-Cas9, which allowed them to induce a DSB at a specific location, the cut site. From this, they observe all the mutations that occurred during repair. The mutations generated are impacted by two key factors: the cell's state (genotype) and the target sequence context [9, 10]. Therefore, Shen et al. used specific target sequence information, such as the nucleotide neighbouring the cut site, as an input into its DNN. With the inclusion of local sequence context, they showed that sequence information is critical for such problems. However, when working with such problems, researchers only consider local sequence context, never the entire sequence. This leads to the question of what additional information can be derived using the complete sequence.

Through advancements in natural language processing (NLP), scientists have repurposed the models created to represent human languages, such as Bidirectional Encoder Representations from Transformers (BERT) [11], and adapted them to DNA sequences. Unfortunately, there is no dictionary of pre-defined words when working with DNA sequences as there is in natural languages. Therefore, scientists format DNA sequences to utilise these models [12]. This is commonly done by treating an entire DNA sequence like a sentence or a document and breaking it down into words. To break a sequence into words, the most common approach is k-mers [12], which generates all the overlapping subsequences of size k for a sequence. For example, from Figure 1, using a $k = 3$, $CTGCA\ldots$ becomes $CTG, TGC, GCA, \ldots$. After generating "words" from DNA sequences, researchers use language models for their prediction problem. This type of research has been increasing in popularity recently, especially with the success shown by DNABERT [13], an adaptation of the BERT architecture [11]. Ji et al. [13] provide pre-trained models using the unlabeled human genome sequences. From these pre-trained models, other researchers can fine-tune using unseen data and tackle different prediction problems.

Formally, this research aims to investigate and evaluate how representations of DNA sequence and mutational spectra can predict NHEJ repair deficiency, as depicted in Figure 2. We hypothesise that a machine learning algorithm can learn intricate relationships between the target sequence representation and the mutational spectra. To achieve this, we answer two research questions: (i) Which categories of mutational spectra most accurately predict NHEJ repair deficiency? (ii) Can sequence representations improve the prediction performance of NHEJ repair deficiency?

We answer our research questions using linear and non-linear machine learning models. We experiment with different categories of mutational spectra and later introduce sequence context. Our analyses are based on the "lib-A" dataset from Shen et al. [6].

2. Methodology

In this section, we describe the methodology used to predict repair deficiencies with mutational spectra and DNA sequence representations. We first describe the data and define our problem formally. Next, we discuss the approach taken towards each subquestion. We start by discussing how we predict the genotype using only mutational spectra. Then, we discuss the different sequence representations used and how these are incorporated with the mutational spectra. We conclude with the experimental setup.

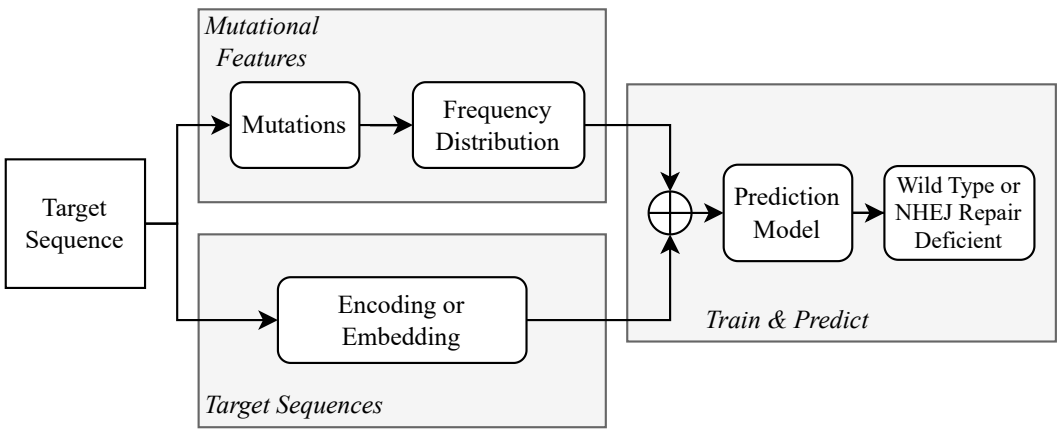


Fig. 2: An overview of our proposed approach. Our proposed approach generates mutational frequency distributions for each unique target sequence in each class. Next, we transform the target sequences into representations using encoding or embedding techniques. Finally, we combine the mutational frequency distribution and target sequence representations to train a model and predict Wild Type or NHEJ repair deficient genotype.

2.1. Data

For our problem, we use the "lib-A" dataset provided by Shen et al. [6]. The dataset contains 1,967 synthetically generated target sequences, which follow the human genome distribution. Each target sequence is made up of 55 nucleotides, and, as shown in Figure 3, there are two crucial attributes. The protospacer adjacent motif (PAM) sequence serves as a molecular marker that helps the Cas9 protein identify the correct location on the DNA strand to cut. The guide RNA (gRNA) is used to locate and bind the DNA sequence to the Cas9 protein, which creates a DSB [14].

Each target sequence in "lib-A" is applied to two cell populations containing mouse embryonic stem cells (mESCs). This allows the researchers to create DSBs and observe mutations in a controlled environment. The first population has the Wild Type genotype, and the other population has the NHEJ repair deficient genotype. The Wild Type genotype represents cells that are not augmented, meaning that all DSB repair pathways should be active. Inversely, the NHEJ repair deficient genotype has two essential genes, the PRKDC and LIG4 genes, knocked out, attempting to impede the use of the NHEJ repair pathway. After creating both cell populations, Shen et al. [6] use CRISPR-Cas9 [8] to create DSBs for each target sequence. Using CRISPR-Cas9, they can create DSBs in specific locations and evaluate the repair product. After introducing a DSB, they let the cells repair themselves and denote all the repair products. These repair products are used to calculate the mutations derived during the repair process. The mutations are calculated by comparing the original target sequence against the repair product. By aligning both sequences at the left-most nucleotide, they can observe any offsets and tabulate the differences (mutations). Shen et al. reported 838,166,630 mutations (count events) under both cell populations and all target sequences.

Our research uses three distinct mutation types: microhomology-less (MH-Less) deletions, microhomology (MH) deletions and insertions, as shown in Figure 4. First, MH-Less deletions are when nucleotides are deleted from the target sequences, and there was no microhomology present at the deletion site (when the DNA strands on either side of the deleted segment align based on the complement). Second, MH deletions are when nucleotides are deleted from the target sequence and at the deletion site, a microhomology was present (underlined nucleotides). Third, insertions are when nucleotides are inserted during the repair process. For this research, we do not consider indel mutations, mutations where an insertion and a deletion occur simultaneously.

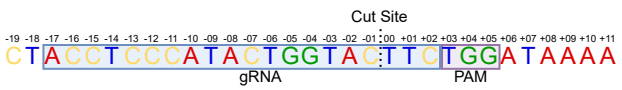


Fig. 3: A fragment of a labelled target sequence from the forward strand. Our target sequences consist of 55 nucleotides, ranging from -27 to +27. The designed guide RNA (gRNA), ranging between -17 and +2, recognises the target sequence. The Cas-9 nuclease makes DSBs at 0, a site 3 base pair upstream to the protospacer adjacent motif (PAM), ranging between +3 and +5.

Fig. 4: Three different mutations from a target sequence. The first row shows a part of the original target sequence. In the first example, the MH-Less deletion, the sequence "GAA" starting at location -2 was deleted, causing the nucleotides on the right to shift towards the other nucleotides. In the second example, the MH deletion, the sequence "AATACG" was deleted. However, this deletion was amended using the sequence's microhomology (underlined nucleotides). Finally, the Insertion example shows the sequence "TC" inserted at the cut site, causing all the nucleotides to the right of it to shift to the right.

2.1.1. Mutation Profiles

The resulting mutational data, called mutation profiles, is described using six attributes, as shown in Table 1. In our context, a mutation profile is the mutations a target sequence observes in a cell population. The **type** attribute indicates whether nucleotides were deleted or inserted. The **size** attribute denotes the number of removed or inserted nucleotides. The **start** attribute indicates the start location of the mutation relative to the original cut site. For example, MH Deletion in Figure 4 had a start of -1, starting one nucleotide to the left of the cut site. The inserted sequence (**InsSeq**) attribute denotes the sequence inserted, which is only applicable for mutations of type insertion. The **homology length** attribute describes the length of the microhomology and only applies to mutations of type deletions. For example, in Figure 4, the MH Deletion has a length of 5. Finally, the **count events** attribute describes how often a specific mutation occurred. Please refer to Appendix A for more complex examples and a deeper analysis of each attribute.

2.1.2. Data Pre-Processing

The resulting mutations might contain mutations caused by external factors. Since we are solely interested in mutations caused by CRISPR-Cas9 DSBs, we opt to pre-process the mutational

Table 1. A sample of the mutation profile for the sequence "GTAAAAGTTAAAATATCTTTAACCTAAAACCGGGTTCAGCGACTTATTTAGTCCA"

Type	Size	Start	InsSeq	Homology Length	Count Events
Deletion	7	-3	N/A	4	3535
Deletion	1	-13	N/A	0	788
Insertion	1	2	A	N/A	1759

data. During this pre-processing, we remove any mutations that appear not to be caused by CRISPR-Cas9. The main filtering condition is based on where the mutations occur [6].

For our study, we only retain mutations that occur at or span the cut site. This means the start location needs to be equal to zero in the case of insertions. When examining deletions, we filter out deletions that do not span the cut site. This condition is met by ensuring that the start location occurs before or at the cut site ($start \leqslant 0$) and the end of this mutation occurs after or at the cut site ($start + size \geqslant 0$). If a microhomology is present, this is taken as part of the deletion. This means that the filtering statement is updated to include the microhomology length as follows ($start - homology\ length \leqslant -1$) and ($start + size \geqslant 0$). For all the exception cases reviewed during this study, please refer to Appendix A.2.

In addition to capturing only mutations caused by a CRISPR-Cas9-induced DSB, we execute two additional checks to ensure that the mutation profile is valid and that the classes remain balanced after pre-processing the mutational data. To ensure the mutational profiles are valid, we only retain a target sequence if the sum of all count events is greater or equal to 100. This removes outliers and exception cases that could negatively impact the prediction. To ensure the classes remain balanced, we retained target sequences with mutations in both genotypes. If a target sequence had mutations in only one genotype, we removed it from the dataset. Following pre-processing, we end with 1,900 target sequences, each observing Wild Type and NHEJ repair deficient mutations. These 1,900 target sequences had 678,814,872 count events, indicating a loss of 19.012% of all count events. To view different filtering examples and more statistical data, please refer to Appendix B.

2.2. Problem Definition

Following the description of the target sequences and their mutational profiles, we describe the problem we are tackling. We make predictions about the genotype of a cell population. For each sequence used to induce a DSB, we use all the mutations obtained from all cells in a population. In addition to this, we hypothesise that this prediction problem can be improved further by introducing details describing the target sequence. Therefore, the problem is extended by including the representations of the target sequence.

Let S be the set of target sequences, and let s_i describe target sequence i, where $i \in [1, \ldots, n]$, and $n = 1,900$, the total unique target sequences. s_i consists of 55 characters, where each character can be made up of A, C, G, or T. Let y be the class variable, where $y \in \{0, 1\}$, 0 describing the Wild Type class, and 1 describing the NHEJ repair deficiency class.

For all s_i, we generate frequency distributions (mutational spectra) based on the mutation profiles as part of the featurisation step (discussed in Section 2.3). The mutational spectra are described using $M \in \mathbb{R}^d$, where d is the dimensionality needed to describe all mutation types arising from all possible combinations of the attributes in Table 1, such as deletion size 1 or homology length 7. Each sample in M describes the mutational frequencies observed under a condition. For example, the mutation frequencies for s_i under the Wild Type genotype. This amounts to 3,800 samples; $n \times 2$ conditions—Wild Type and NHEJ repair deficient.

As for the target sequences, we generate sequence representations, such as k-mer frequency, to describe the sequence information (discussed in Section 2.4). For all s_i, a representation r_i is generated, where $r_i \in \mathbb{R}^l$, and l describes a fixed number of dimensions. We denote R as the matrix of all S representations with a dimensionality of $1,900 \times l$.

Following the definition of M and R, we define the feature vector X. For the first research question, we use only mutational spectra, $X = M$. In this case, each feature describes a unique mutation type, and each sample describes the condition this was observed under (s_i and genotype). Afterwards, to tackle the second research question (using both mutational spectra and target sequences), $X = M \oplus R$, where the R representations are concatenated with the M vectors[1]. This feature vector results in $X \in \mathbb{R}^{d+l}$.

Finally, for the feature vector X, we want to develop a model $f(x)$. This model accepts the feature vector X, describing the mutational spectra with/without the target sequence representation. Using this feature vector, the model predicts the binary class labels y describing the state of the cell, Wild Type or NHEJ repair deficient, as depicted:

$$\left[\begin{array}{c|cccccccccc}
y^1_{wt} & x^1_{1_{wt}} & \ldots & x^1_{j_{wt}} & \ldots & x^1_{d_{wt}} & r^1_1 & \ldots & r^1_k & \ldots & r^1_l \\
y^1_{rd} & x^1_{1_{rd}} & \ldots & x^1_{j_{rd}} & \ldots & x^1_{d_{rd}} & r^1_1 & \ldots & r^1_k & \ldots & r^1_l \\
\vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\
y^i_{wt} & x^i_{1_{wt}} & \ldots & x^i_{j_{wt}} & \ldots & x^i_{d_{wt}} & r^i_1 & \ldots & r^i_k & \ldots & r^i_l \\
y^i_{rd} & x^i_{1_{rd}} & \ldots & x^i_{j_{rd}} & \ldots & x^i_{d_{rd}} & r^i_1 & \ldots & r^i_k & \ldots & r^i_l \\
\vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\
y^n_{wt} & x^n_{1_{wt}} & \ldots & x^n_{j_{wt}} & \ldots & x^n_{d_{wt}} & r^n_1 & \ldots & r^n_k & \ldots & r^n_l \\
y^n_{rd} & x^n_{1_{rd}} & \ldots & x^n_{j_{rd}} & \ldots & x^n_{d_{rd}} & r^n_1 & \ldots & r^n_k & \ldots & r^n_l
\end{array}\right]$$

where y^i_{wt} and y^i_{rd} denote the Wild Type or NHEJ repair deficient labels, respectively. $x^i_{j_{wt}}$ describes the frequency of mutation j of the Wild Type class for the of s_i, and $x^i_{j_{rd}}$ describes the frequency of mutation j of the Repair Deficient class for the of s_i. In addition, r^i_k describes the representation of target sequence i at dimension k.

2.3. Mutational Spectra Features

Following the problem definition, we look into the first research question: "Which categories of mutational spectra most accurately predict NHEJ repair deficiency?" The mutational spectra allow us to describe all the mutations observed. This approach can handle common challenges of mutational data, such as varying mutation rates among different DNA sequences. Different DNA sequences are prone to generating different amounts of mutations [15], making it difficult to compare them. Summarising the mutational outcomes as frequency distributions along specific attributes, such as deletion size or microhomology length, gives the flexibility to evaluate these different attributes with respect to how they help the model perform on the prediction task.

The mutations depend on the target sequence and the repair pathway. Therefore, we expect to observe differences between the general distribution trend observed under the Wild Type and NHEJ repair deficient class. In addition to this, we expect to observe some differences between the distributions generated for different target sequences under the same class.

[1]The same representation is added to both the Wild Type and NHEJ repair deficient samples.

2.3.1. Featurisation

In this study, we utilise the mutational spectra rather than the mutation profiles discussed in Table 1. For our research, we calculate the frequency distributions of deletion sizes, insertions, and homology lengths. These three categories were selected because they showed the most frequency variation between both classes. To generate the frequency distribution for each category, we calculate the normalised count such that the values add up to 1 if the category was observed for the target sequence and 0 otherwise.

The procedure of generating the frequency distribution starts by transforming all of the mutational attributes into features. For example, in the case of deletion sizes, the features become deletion size $1, 2, \ldots, 40$, where their values are equal to the count events reported by a target sequence under a cell population. After this featurisation, the values are normalised by dividing by the sum of the count event for the category in a target sequence under a cell population. This results in the frequency distribution features for one category. For example, the first row of Figure 5 describes the frequency distributions of target sequence AT...CG under the cell population Wild Type, 36.77% of all deletions had size 1. The second row describes the frequency distributions of the same target sequence, but under the cell population NHEJ repair deficient, 18.39% of all deletions had size 1.

When using multiple categories together, we extend the frequency distributions by concatenating them, as depicted in Figure 5. This is done by first calculating the frequency distribution for each category and then joining them together. This means if there are two categories (M_1, M_2), for example, deletion size and insertion, $M_1 = \{x_1, x_2, \ldots, x_{d_1}\}$, where $x_i \in [0, 1]$ and $i \in [1, \ldots, d_1]$; $M_2 = \{u_1, u_2, \ldots, u_{d_2}\}$, where $u_j \in [0, 1]$ and $j \in [1, \ldots, d_2]$. The resulting feature vector is $X \in \mathbb{R}^{2n \times (d_1 + d_2)}$, where n is the number of sequences, each with two possible genotypes (Wild Type/NHEJ repair deficient), and d_1 and d_2 are the number of unique mutations from M_1 and M_2, respectively.

We opted to use the frequency of mutational features to address the mutation rate issue mutational data have [15, 16]. Seeing as different target sequences observed different types of mutations all with varying rates of occurrence, these need to be normalised. These frequency distributions treat all the samples

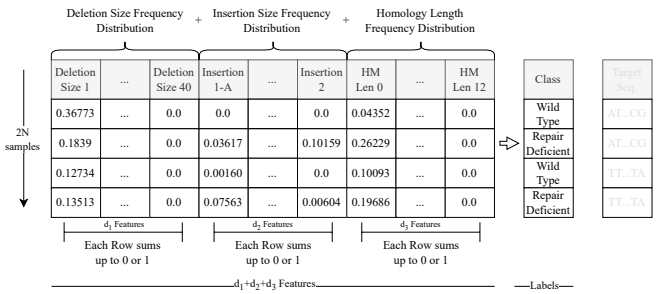


Fig. 5: The mutational frequency features visualised and labelled. The visualization is organised into rows $(2N)$ representing the samples and columns $(d_1 + d_2 + d_3)$ representing the features. To create the features, we generated frequency distributions of categories we are interested in and concatenated them. For a listing of the features that make up each category, please refer to Appendix C.

with equal importance. This means that if the Wild Type class saw more count events than the NHEJ repair deficient class, the frequency distributions will not retain such information. Instead, it will describe the frequencies each sample observed, highlighting the importance of certain features over others. This allows us to treat both samples in the input matrix as equally important. In addition, the featurisation allows for multi-distribution comparison, where one or more distributions from different categories can be used together to describe a single set of mutation profiles. Multi-distribution allows for the possibility of including as few or as many categories as necessary for the prediction problem at hand.

2.3.2. Distributions and Combinations

We use the term "frequency distribution groups" to discuss different category combinations. A frequency distribution group describes either a frequency distribution of a single category or multiple frequency distributions concatenated for multiple categories. In this study, we use six different frequency distribution groups.

Mutation Type (MT) Frequencies (3 features): whether the mutation represents an insertion, a MH, or a MH-less deletion.

Homology Length (HL) Frequencies (13 features): homology lengths, ranging from -1 to 12, excluding homology length 10 (never observed), with -1 describing the insertions, 0 for MH-Less deletions and 1-12 for each homology length. For this specific group, insertions are encoded with homology lengths since there were instances where a target sequence only reported insertion mutations.

Deletion Size and Insertion Frequencies (DS-Ins) (45 features): The deletion size frequency, 40 features, for deletion sizes between 1 and 40. And the Insertion Frequency, five features (1-A, 1-C, 1-G, 1-T, for insertions of size one and insertions of size 2+), where larger insertion sizes are combined into a single feature due to the sparse nature of these features (as shown in Appendix A.1).

Deletion Size, Insertion, and Homology Length Frequencies (DS-Ins-HL) (57 features): the same structure as DS-Ins, but it includes Homology Length frequencies within its features, ranging from 0-12, 0 for MH-Less deletions and 1-12 for each homology length. The previously described insertion feature (-1) in homology length frequencies is removed, as this information is now encoded using the five insertion features.

Split Deletion Size and Insertion Frequencies (SDS-Ins) (84 features): the insertion frequency distribution (5 features), the MH-Deletion size frequency, 40 features, for MH-Deletion sizes between 1 and 40, and the MH-Less deletion size frequency, 39 features, for deletion sizes between 1 and 39.

Split Deletion Size, Insertion and Homology Length Frequencies (SDS-Ins-HL) (96 features): the same structure as SDS-Ins, but it includes Homology Length frequencies within its features, ranging from 0-12, 0 for MH-Less deletions and 1-12 for each homology length. The previously described insertion feature (-1) in homology length frequencies is removed, as this information is now encoded using the five insertion features. This deletion-type split (SDS-Ins and SDS-Ins-HL) is primarily due to the MMEJ repair pathway. This pathway utilises the microhomologies within the deletion to repair the DSB. Considering this information, we wanted to capture more subtle differences between both classes, as the MMEJ pathway should be more active when NHEJ is deficient.

To achieve all the frequency distributions mentioned, we concatenate different distributions, as shown in Figure 5, giving us the resulting feature space. This type of combination allows us to evaluate the impact of different distributions—for example, the impact of deletion size distribution with and without the homology length distribution.

2.3.3. Feature Binning

As the frequency distribution features represent all the possible observable mutations, we must handle features with high-zero count. When working with distribution frequencies, uncommon mutations, such as deletion size 39/40, are rarely observed, resulting in high zero-count values (as shown in Appendix A.1). Due to their rare nature, these types of mutations could impede the model from generalising to an optimal solution. We opt to enforce a binning technique on the frequency distributions to cater to this and any previously unseen mutations, such as deletion size 50.

We chose percentile binning over a fixed-width binning technique. Whereas fixed binning combines features without considering the data distribution [17], percentile binning only combines the features if they have a high-zero count [18], avoiding unnecessary information loss. When using a percentile technique, a variable n is chosen. This variable determines the percentage of samples in each feature with a non-zero value. If more than $n\%$ of the data has a zero value, this technique combines the next features (in ascending order) until the condition of $n\%$ data samples with a non-zero value is met, or all the features are exhausted [18].

This technique is applied to the frequency distribution of each category independently. For example, in the case of DS-Ins-HL, we calculate the binning percentile for the deletion sizes and the binning percentile for the homology length. Afterwards, we use the resulting binned features as the feature set. Insertion frequencies are not binned since these are already pre-binned (2+) during the pre-processing step.

After experimentation, we chose a 1% binning strategy. This percentile value allowed us to retain a good resolution on small and medium-sized mutations while reducing the sparsity in infrequent mutations. The resulting feature spaces were transformed, resulting in the following dimensions: MT—3 features; HL—10 features; DS-Ins—30 features; DS-Ins-HL—39 features; SDS-Ins—55 features; DS-Ins—64 features. For an in-depth analysis of different binning percentile values and the resulting combined features, please refer to Appendix D.

Following the frequency distribution generation, different frequency grouping, and feature binning, we end up with six different frequency distribution groups that should distinguish between Wild Type and NHEJ repair deficient mutations.

2.4. Target Sequence Representation

After discussing the mutational features, we look into including target sequence information to predict Wild Type and NHEJ repair deficiencies, the second research question. We hypothesise that a machine-learning model could identify interactions between mutational spectra features and sequence representations, which impact the resulting class. For example, Molla et al. [19] found that the NHEJ repair pathway is prone to creating one nucleotide insertions, commonly identical to the nucleotide at -1.

For a machine to interpret and understand a target sequence, it must be represented numerically. There exist various methods that transform a DNA sequence into a numeric representation. In the following subsections, we discuss two approaches: encoding and embedding the target sequences. By investigating both approaches, we aim to understand if a simple encoding is enough to observe an improvement from the mutational spectra features or if a more complex representation, an embedding, is needed to capture meaningful information.

Sequence encodings provide a direct translation of the sequence, that is, a human-understandable representation. When using these approaches, the generated representations are often simple and easy to interpret. However, their basic nature makes capturing higher-level relationships in the sequence challenging and often impossible [20].

Sequence embeddings, seen as a sequence transformation, aim to capture patterns that provide higher-level semantic information. These representations are often computationally more intensive than encoding techniques, and the resulting representation is only human-understandable with additional tools or techniques [12, 21].

2.4.1. Encoding Representations

One commonly used encoding technique is One-Hot Encoding (OHE). When using OHE for DNA sequences, each unique nucleotide is represented using a unique binary vector representation, for example, ($A = [1\ 0\ 0\ 0]^\intercal$; $C = [0\ 1\ 0\ 0]^\intercal$; $G = [0\ 0\ 1\ 0]^\intercal$; $T = [0\ 0\ 0\ 1]^\intercal$;). Using these vectors to represent the sequence "GCA", the resulting encoding is the binary matrix representation $[[0\ 0\ 1\ 0]^\intercal\ [0\ 1\ 0\ 0]^\intercal\ [1\ 0\ 0\ 0]^\intercal]$.

While this approach captures positional information about the nucleotides, the resulting representation is sparse, requiring 220 features to describe a 55-nucleotide sequence. This sparsity and dimensionality scaling directly with the sequence length makes it burdensome to scale for long target sequences. This prompted the investigation into a different type of sequence encoding, the k-mer frequency encoding.

k-mer frequency encoding captures the frequency of each possible k-mer in a target sequence. Initially, it extracts all the k-mers present in a sequence using a sliding window of size k and a stride of one (i.e., capture a "word" made up of k-characters, move one character to the right and repeat until all the characters are exhausted). For example, using a $k = 3$, $CTGCA\ldots$ becomes $CTG, TGC, GCA, \ldots$. After all the k-mers are extracted, it calculates the frequency of each k-mer in that sequence. These frequencies are then recorded in the dictionary of k-mers. The dictionary of k-mers is a feature vector of (4^k), describing all the possible k-mers of size k. For example, using a $k = 1$ to calculate the k-mer frequency of the sequence "$ACATG$" (which can be considered as nucleotide frequency counting), the resulting frequency would look like $[0.4, 0.2, 0.2, 0.2]$. Where the elements describe the frequencies of A, C, G, T, respectively.

When dealing with small k values, k-mer frequency encoding is a less sparse representation than one-hot encoding. However, k-mer frequency encoding does not describe any positional information about the nucleotides, and there is an exponential growth in the feature space based on the k value. For example, when $k = 2$, the dictionary size is 16 k-mers $(AA, AC, AG, AT, CA...)$, whereas when $k = 3$, the dictionary size is 64 k-mers $(AAA, AAC, AAG, AAT, ACA...)$. This study uses a $k = 1$ for the k-mer frequency encoding because it yields interpretable representations and little to no sparsity. For a review of the $k = 2$, please refer to Appendix E.1.

2.4.2. Embedding Representations

Embedding representations transform the input by learning relationships between different parts of the sequence. With the advancements in natural language processing (NLP), researchers are adopting these higher-complexity models to allow for more descriptive representations of DNA sequences [12]. These models commonly use functionality, such as self-attention, to capture short and long interactions between different k-mers ("words") [12]. The self-attention mechanism computes attention scores between all pairs of input tokens, allowing the model to capture local and global dependencies [22]. Using representations that can capture holistic information, we hypothesise that some machine-learning algorithms can benefit from introducing such features.

To generate the embedding representations, we use DNABERT, a pre-trained Bidirectional Encoder Representations from Transformers (BERT) model for DNA sequence [13]. BERT is a transformer-based language representation model that performs well in many NLP tasks [11]. This model uses pre-training and fine-tuning, where it initially learns a general-purpose understanding using large amounts of unlabelled data. Then, during fine-tuning, it solves various applications with task-specific data. DNABERT is a specialised variant of the BERT model explicitly designed for embedding DNA sequences. Unlike the BERT architecture, DNABERT does not use the next-sentence prediction functionality since target sequences (analogous to "sentences") are not expected to aid in predicting the next sequence, unlike natural language. Similarly to the BERT model, DNABERT has two phases: pre-training and fine-tuning. To describe sequences, the model transforms sequences into numeric representations of overlapping k-mers with a position-dependent vector. As a deliverable of their research, Ji et al. [13] published four pre-trained models (using k values of 3, 4, 5, and 6 during the tokenisation process).

We opted to use DNABERT for the embeddings primarily for two reasons. Firstly, the target sequences are synthetically generated so that they can generalise well to arbitrary sequences from the human genome [6]. This means that the learned DNABERT embeddings, which is trained on the human genome, should represent the sequences. Secondly, DNABERT was trained and evaluated on sequences of lengths from 5 to 512 nucleotides, allowing small sequences to be represented effectively. This aligns with the target sequences, which are comprised of 55 nucleotides.

When working with DNABERT, the embedding representations are made up of two components: the sentence-level classification and the token-level classification. The sentence-level classification token, with a dimensionality of (1×768), is optimised during fine-tuning and is commonly referred to as the summary of all the tokens in a sentence. On the other hand, the token-level classification tokens are trained during pre-training and optimised during fine-tuning; these tokens describe each token individually, and through the use of the attention layers, they embed information describing all the other tokens [13]. Each token in the token-level classification has a dimensionality of (1×768), resulting in a dimensionality of $(t_k \times 768)$ when considering all of the tokens. For a sequence of length L and a k-mer of size k, t_k describes the total number of k-mers generated, using $t_k = L-k+1$ (for a sequence of length 55 and a k-mer of 3, $t_k = 53$). For this research, we focused on the token-level classification embeddings, as these should contain the highest level of information. Also, since the model is not fine-tuned, the sentence-level classification might not capture all the necessary information. Our research focuses on the pre-trained DNABERT model, with a $k = 3$, denoted as DB3. For the different k values, please refer to Appendix F.

We use token-level representations with a high-dimensional space of 40,704 dimensions for a single sequence; with $k = 3$, $L = 55 \rightarrow (53 \times 768)$. Unfortunately, with a limited dataset of only 1,900 sequences (or 3,800 samples), this space needs to be reduced due to the curse of dimensionality [23]. Therefore, we propose three dimensionality reduction techniques: mean pooling, horizontal stacking and vertical stacking; to transform these representations into an appropriate size. This reduction allows the transformed embeddings to be used in collaboration with the mutational spectra features. To formally define the reduction techniques, the following variables are used: let n be the number of sequences, k be the k-mer size, t_k be the number of tokens when using the k-mers, and d be the number of dimensions. We use $n = 1900$, $k = 3$ and $t_k = 53$ in the provided figures.

Starting with **mean pooling** (MP), this technique, for each target sequence, generates a token representing the average values of all the token embeddings. As depicted in Figure 6, we start with a dimensionality of $(n \times t_k \times 768)$. We average over all the $(t_k \times 768)$ to end with a dimensionality of $(n \times 768)$. When using this technique, we have a simple yet descriptive approach to all the tokens [24]. The original space can never be restored when using this type of reduction.

When using **horizontal stacking** (HS), as depicted in Figure 7, it starts by stacking all the tokens horizontally. So for each target sequence, which has a representation of $(t_k \times 768)$, the feature space is flattened out from a three-dimensional space to a two-dimensional space; $(n \times 53 * 768)^2 \rightarrow (n \times 40,704)$. After this representation is flattened, a dimensionality reduction technique such as PCA [25] or UMAP [26] is deployed. This technique reduces the 40,704 into a smaller dimension, such as 100, resulting in a dimensionality of $(1,900 \times 100)$. This approach

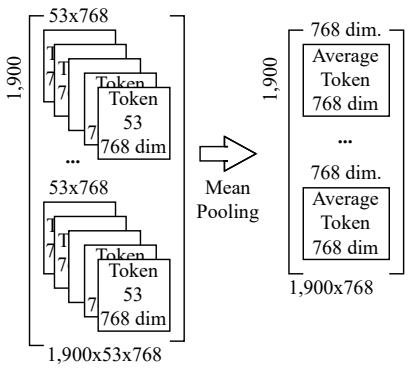


Fig. 6: A visualisation of the mean pooling technique for the DNABERT token embeddings. We start with a dimensionality of $(1,900 \times 53 \times 768)$. For each target sequence (1,900), we obtain the average values for all the token embeddings. This means that for each sequence, the tokens are averaged on (53×768), resulting in a dimensionality of (1×768). After all the sequences had their token embeddings averaged, the resulting dimensionality is $(1,900 \times 768)$, where each row represents a sequence, and the columns represent the average token-level embedding.

[2] "$\times$" denotes the dimensionality; "$*$" denotes stacking

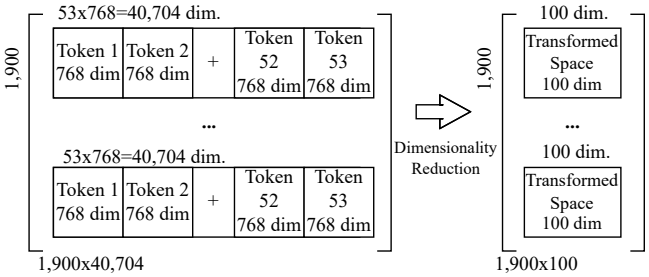


Fig. 7: A visualisation of the horizontal stacking technique for the DNABERT embeddings. First, the original space $(1,900 \times 53 * 768)$ is stacked horizontally by appending all the token embeddings that belong to a target sequence together. This creates a new dimensionality of $(1,900 \times 40,704)$. Next, it performs dimensionality reduction on the 40,704 features using a suitable technique such as PCA or UMAP. This results in a new dimensionality of $(1,900 \times 100)$, where all the token embeddings have been reduced to 100 dimensions for each sequence.

aims to retain the information learned during the embedding procedure, such as global context information, but represent it in a smaller dimensionality [27]. The resulting transformed space is not interpretable without additional steps to transform it back into the original space.

Finally, we developed **vertical stacking** (VS), as shown in Figure 8. This approach allows the interpretation of tokens based on positional information. When using this reduction approach, all tokens are stacked vertically. This means that each token, which has a representation of $(t_k \times 768)$, is stacked on top of each other. This transforms a three-dimensional space into a

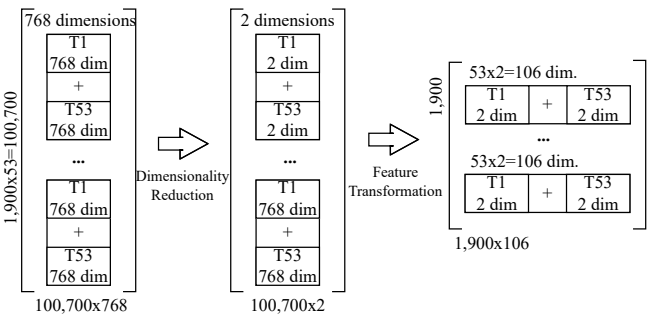


Fig. 8: A visualisation of the vertical stacking technique for the DNABERT embeddings. First, the original space $(1,900 * 53 \times 768)$ is stacked vertically by appending all the token embeddings together, keeping the order from which they occur but ignoring the target sequence from which they are derived. This creates a new dimensionality of $(100,700 \times 768)$. Next, we reduce dimensionality along the 768 features using a suitable technique such as PCA or UMAP. This results in a new dimensionality of $(100,700 \times 2)$, where the token embeddings have been reduced to 2 dimensions for each token. Finally, the resulting dimensionality is transformed and returned to the original space $(1,900 \times 53 \times 2)$, where each row represents a target sequence and the columns represent a 2D embedding for a particular token.

two-dimensional space $(n * t_k \times 768) \rightarrow (t_k(n) \times 768)$, losing reference to the neighbouring tokens. After this transformation, we employ a dimensionality reduction technique to reduce the 768 dimensions into a smaller one, for example, two dimensions per token, resulting in a dimensionality of $(t_k(n) \times 2)$. Once complete, the tokens are rearranged into their original position $(n \times t_k \times 2)$.

Even though this approach allows for the resulting feature space to be interpreted based on the token position, it has some disadvantages due to the concatenation step. Since tokens for all sequences are stacked into a single space and then transformed, each token is represented using only a fixed dimension size (two in Figure 8). This might be problematic since all tokens have the same number of dimensions, even though not all tokens may be equally important. In addition, during the space reduction, the target sequence context is likely to be lost. This is because the space does not consider the complete target sequence but does the space reduction based on all the tokens in the space, irrespective of which sequence originated which token.

Following the description of the mutational frequency embeddings and the embedding of the sequences, we look at the experiments and evaluations needed for our proposed approach.

2.5. Experimental Setup

We use the remaining 1,900 target sequences after pre-processing to address our research questions: (i) Which categories of mutational spectra most accurately predict NHEJ repair deficiency? (ii) Can sequence representations improve the prediction performance of NHEJ repair deficiency? Following the featurisation step, each target sequence has two samples, one for each class, resulting in the input matrix having 3,800 samples. We instantiate the NHEJ repair deficiency as the positive class and the Wild Type as the negative class.

The train and test sets are generated using an 80:20 split. This results in 1,520 sequences/3,040 samples for the train set and 380 sequences/760 samples for the test sets. In addition to the train:test split, we use 10-fold cross-validation on the training set, resulting in training and validation sets of 152 sequences/304 samples in each fold. This allows us to make informed decisions about the models while remaining impartial about the test set. To avoid data leakage from either set or fold, we ensure that each target sequence only belongs to one set and one fold, meaning that both samples of a target sequence are always paired together.

For learning and predicting this binary classification problem, we employ two machine learning algorithms. The first algorithm, Logistic Regression (LR), shows us how a linear model can predict the genotype. The logistic regression allows for easily interpretable features. The second algorithm, Random Forest (RF), can capture non-linear relationships between sequence representations and mutational features, possibly addressing the shortcomings of the Logistic Regressor. Before using the Logistic Regression, we standardise the features by removing the mean and scaling to unit variance. This is not applied to the Random Forest as this is based on tree partitioning algorithms. Therefore, there is no analogue to a coefficient found in general regression strategies [28]. Both algorithms and feature scaling are implemented using the Scikit-Learn library [29].

As discussed in Section 2.3.2, we use six distribution groups to evaluate which categories of mutational spectra most accurately predict NHEJ repair deficiency. After evaluating these distribution

groups, we add the target sequence representations to the best-performing group to study their effect.

When evaluating the performances of these models, we report the accuracy score since the classes are balanced, where both classes have the same amount of samples. All the scores described in Section 3 are derived from the cross-validation scores. The performance scores for the held-out samples (test set) are described in Appendix E.7. To review the additional metrics collected (F1-Score, precision, recall, area under the precision-recall curve (AUPRC), and area under the receiver operating characteristic (AUROC)), please refer to Appendix E.6.

In addition to these scores, we look into the odds ratio and feature permutation importance. These two other metrics describe the learned model feature weights, allowing us to understand the effect of different features better.

The Odds Ratio is a statistical measure commonly used in logistic regression to quantify the strength and direction of the relationship between a binary outcome variable and one or more predictor variables [30]. This measurement aids in understanding how the odds of an event occurring change with a one-unit change in a predictor variable while keeping other variables constant. Mathematically, this is calculated using e^{β}, where β is the coefficient associated with a predictor variable in a logistic regression model. In our context, since NHEJ repair deficient is the positive class, the odds ratios are interpreted as follows:

- $Odds > 1$: A one-unit increase in the feature coefficient increases the odds of an event belonging to the NHEJ Repair Deficient class. This indicates a positive association between the feature and the positive class.
- $Odds = 1$: A one-unit increase in the feature coefficient does not change the odds of an event belonging to the NHEJ Repair Deficient class. This indicates no association between the feature and the positive class.
- $Odds < 1$: A one-unit increase in the feature coefficient decreases the odds of an event belonging to the NHEJ Repair Deficient class. This indicates a negative association between the feature and the positive class.

Feature Permutation Importance (FPI) [28, 31] is another technique used to assess the importance of individual features in a machine-learning model. It helps us understand how much a feature contributes to the model's predictive performance. After training and scoring the model, FPI is initiated, where it proceeds to evaluate the features. This process is done by selecting a feature, randomly shuffling the feature across all data points, and then evaluating the model's performance with the shuffled feature. The decrease in the model's performance compared to the original score measures how much the feature contributed to the model's accuracy. The larger the decrease, the more essential the feature.

When comparing the results against each other, we use the Wilcoxon signed-rank test. This is a nonparametric test equivalent to the paired samples t-test. In our case, we pair the result of Validation Fold 1 in Experiment 1 with Validation Fold 1 in Experiment 2, a total of 10 pairs. The data samples seen by each fold are constant throughout the experiments and runs. A nonparametric test is used because we have a limited sample size. Therefore, we cannot ensure that the data is normally distributed.

We use Plotly [32] to visualise our results and generate all the figures in Section 3. To visualise these figures in an interactive and scalable manner, please visit *https://thesis.jborg.dev/*.

3. Results & Discussion

To evaluate the proposed approach, we split the evaluation into two phases, aligning with our research question. Firstly, we investigate the mutational spectra features. By understanding the most influential and best-performing categories, we ensure that we use the most informative distributions. Secondly, we analyse how the target sequence representations aid the mutational spectra features and determine if the models can learn additional details, which improve upon the mutational spectra accuracy.

Target sequences impact the selection of repair pathways and mutation frequency distributions generated. Therefore, we hypothesise that sequence information can introduce beneficial information that aids the prediction problem. That is, predicting Wild Type and NHEJ repair deficient genotypes.

3.1. Mutational Spectra Features

As discussed in Section 2.3.2 and summarised in Table 2, we generated six frequency distribution groups to understand the impact of combining different categories. The groups describe the mutations observed at different levels of detail, ranging from MT, which uses only three features, to SDS-Ins-HL, which includes the most granular features.

In Figure 9, the random forest model (RF) vastly outperforms the logistic regression model (LR) for all the distribution frequency groups used. Even the worst RF model does almost as well as the best-performing LR model. We hypothesise this is attributed to two key factors: the ability to capture non-linear relationships and the bagging functionality of this algorithm.

RF is capable of capturing non-linear relationships among the different features. As the features describe different frequency distributions, a linear model struggles to capture these interactions between different mutational frequencies, for example, between different deletion sizes. The potential non-linearity relationship is further amplified when using multiple categories (such as DS-Ins-HL) in the frequency distribution group. For example, the relationship between the individual deletion size and homology length features might be too complicated to capture in a linear space.

Table 2. The frequency groups each experiment uses to train and predict the respective models.

	Mut. Type Freq.[1]	HM Len. Freq.[2]	Ins. Freq.[3]	Del. Size Freq[4].	Del. Size Freq. Split by MH/ MH-Less[5]
MT	✓				
HL		✓			
DS-Ins			✓	✓	
DS-Ins-HL		✓	✓	✓	
SDS-Ins			✓		✓
SDS-Ins-HL		✓	✓		✓

[1]Mutation Type Frequency

[2]Homology Length Frequency Distribution

[3]Insertion Frequency Distribution

[4]Deletion Size Frequency Distribution

[5]Deletion Size Frequency Distribution split by MH/MH-Less

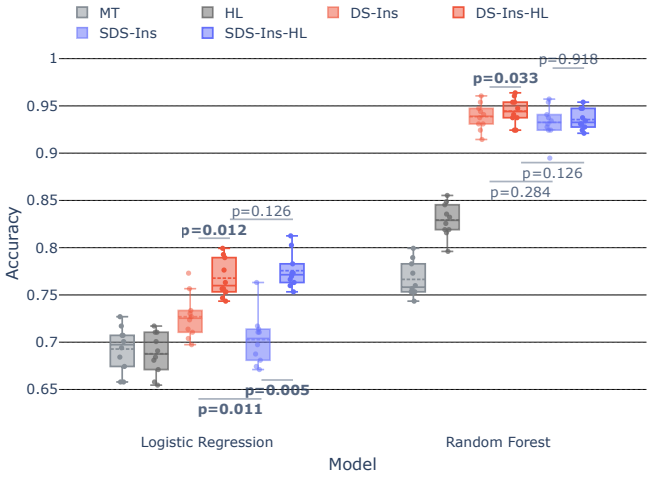


Fig. 9: Validation accuracy scores for all different feature groups (`url`). MT—Mutation Type; HL—Homology Length; DS—Deletion Size; Ins—Insertion; SDS—Split Deletion Size.

In addition, RF uses bagging to combine the prediction of several learners, allowing it to better generalise the data. This functionality draws multiple samples randomly (with replacement) from the original dataset to create new training sets. Bagging helps reduce the variance of the model, and as shown from the RF results, there is a notable difference in the variance reported between both models. Furthermore, RF also introduces randomness in the feature selection process. Instead of using all features to split a node, only a random subset of features is considered. This helps to reduce the correlation between the trees and improve the diversity of the forest. By having a diverse forest, the algorithm can describe more cases. Since mutational spectra are dependent on the target sequence, this diversity is better suited since the LR might struggle to find a consensus which can describe the overall composition of the frequency distribution group.

3.1.1. Understanding Different Frequency Groups

As depicted in Figure 9, DS-Ins (LR: 72.70% ± 2.34%; RF: 93.85% ± 1.39%) and DS-Ins-HL (LR: 76.78% ± 2.02%; RF: 94.44% ± 1.39%) yielded some of the best-performing distribution groups when using either model. This type of performance suggests that the models are learning to discriminate between both classes using the deletion sizes, as previously hypothesised.

To further evaluate these scores, we cross-compare the related experiments. This first cross-comparison is done on DS-Ins vs. SDS-Ins and DS-Ins-HL vs. SDS-Ins-HL. These two sets are meant to allow us to evaluate the impact of splitting the deletion size category by deletion type. Secondly, DS-Ins vs. DS-Ins-HL and SDS-Ins vs. SDS-Ins-HL are meant to allow us to evaluate the impact of the homology length category on the performance score.

Since deletion size frequencies were critical for the prediction problem, we hypothesised that MH deletion size frequencies and MH-Less deletion size frequencies could further refine the prediction score. The deletion size frequency split shows that the models cannot add additional useful information that aids with the prediction problem. SDS-Ins (deletion size frequencies split by MH/MH-Less and insertion frequencies) reported a score of LR: 70.26% ± 2.72% and RF: 93.26% ± 1.75%. This showed a significant drop ($p = 0.011$) in LR compared to DS-Ins (deletion size and insertion frequencies). In contrast, SDS-Ins-HL reported no significant changes when compared to DS-Ins-HL with accuracy scores of LR: 77.57% ± 1.88%, $p = 0.126$ and RF: 93.26% ± 1.75%, $p = 0.126$.

Homology length is a key distribution when classifying the two genotypes. When comparing DS-Ins vs. DS-Ins-HL, LR reported a significant increase (72.70% ± 2.34% → 76.78% ± 2.02%, $p = 0.012$). Similarly, RF reported a significant increase (93.85% ± 1.39% → 94.44% ± 1.39%, $p = 0.033$). Similar trends followed when comparing SDS-Ins and SDS-Ins-HL; the LR reported a significant increase (70.26% ± 2.72% → 77.57% ± 1.88%, $p = 0.005$), but RF saw no changes in performance.

Finally, when using the most simplistic features, both MT and HL struggled to achieve the performance of other feature sets for both LR and RF models. MT obtained the lowest recorded accuracy scores during cross-validation (LR: 68.28% ± 2.38%; RF: 76.64% ± 1.86%). In comparison, HL obtained the second lower scores (LR: 68.78% ± 2.21%; RF: 82.93% ± 1.78%). This difficulty in predicting genotypes suggests that the distributions did not encode sufficient information to make informed predictions.

Based on these findings, we use DS-Ins-HL (deletion size, insertion, and homology length frequencies, per Section 2.3.2) as our designated frequency group. Apart from achieving some of the best-performing scores reported with either model, there is no significant difference ($p = 0.126$) from the second best-performing distribution group, SDS-Ins-HL. SDS-Ins-HL uses a more granular frequency distribution, meaning a simpler feature set should allow for easier interpretation.

3.1.2. Deletion Size and Homology Length Frequencies

This prompts us to investigate which features in DS-Ins-HL are more deterministic of the output classes by looking into this feature group's odds ratio and FPI.

Figure 10, depicting the odds ratios of DS-Ins-HL, shows that small deletion size and small homology length features are more discriminative than larger sizes/lengths. These odds ratios show that small deletion sizes strongly correlate with the negative class (Wild Type), reinforcing the understanding that NHEJ (active in the Wild Type genotype) generates short deletions. Medium/long deletion sizes are not very discriminative of the output class, with only the largest deletion sizes (29+) being positively associated with the prediction of NHEJ repair deficiencies. We hypothesise that the deletion of size one was less influential than other small deletion sizes, primarily due to staggered cuts. Staggered cuts are when the DSB is not broken perfectly, leading to one strand breaking at a different position than the other strand. Due to these imperfections in the DSB, staggered cuts could have leaked into the data set and been classified as a mutation during the data collection process, leading to a less discriminative feature than small deletion sizes.

Focusing on insertion frequencies, one nucleotide insertions are more discriminative than larger insertions (2+), with both features negatively associating with the NHEJ repair deficient class. A closer look at the one nucleotide insertions reveals that A's and T's are more negatively associated than C's and G's. These findings can be linked with other findings and the composition of the target sequences. Molla et al. [19] found that mutation outcomes of the NHEJ repair pathway are most reliably predicted using one nucleotide insertions. These insertions are commonly identical

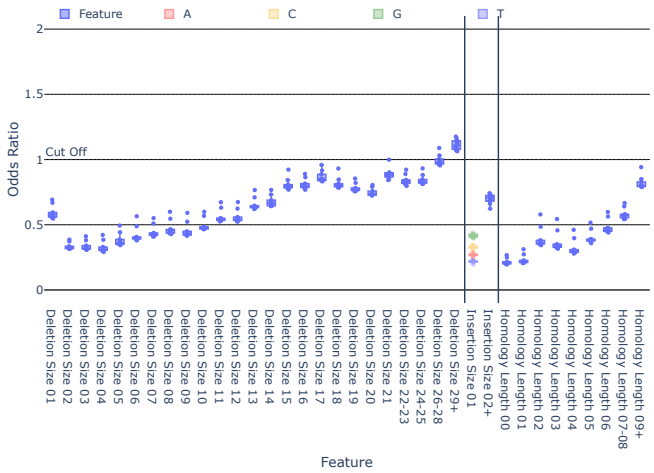


Fig. 10: The odds ratio values for the logistic regressio (LR) coefficients of DS-Ins-HL — Deletion Size, Insertion, and Homology Length Frequencies (`url`).

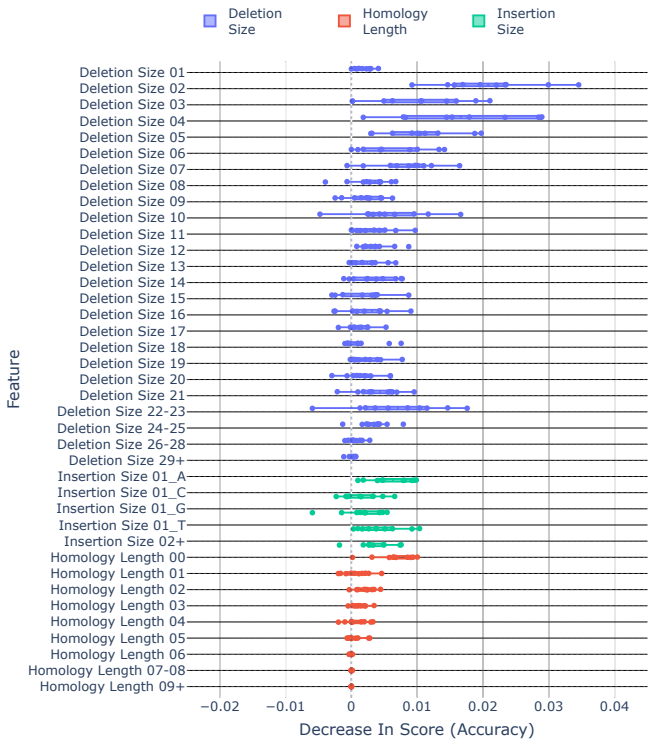


Fig. 11: The feature permutation importance on the validation set for DS-Ins-HL — Deletion Size, Insertion, and Homology Length Frequencies (`url`). The value is averaged over 20 random shuffles.

to the nucleotide at -1 (-4 from the PAM). This suggests that the LR model could identify the class using similar information. Furthermore, the reported odds ratio difference can be attributed to two factors. Molla et al. [19] identified that the predictability decreases in the order $T > A > C > G$, aligning with our odds ratio findings. Additionally, as discussed in Appendix A.1, the dataset's most common nucleotides at -1 are A's and T's, possibly contributing to the odds ratio values.

Lastly, the homology length frequencies show a similar trend to the deletion size frequencies. Feature homology length 00 (indicating no microhomology) is the most impactful, with an odds value averaging 0.2. As expected, this has a negative association with the NHEJ repair deficiency class, as the NHEJ repair pathway should only be active in the Wild Type class, meaning that more microhomology-less deletions should be seen in that class. Larger microhomology lengths are less discriminative of the prediction class, indicating that these lengths could occur in both classes but are more favourable towards the Wild Type label.

Focusing on the FPI for the RF model, shown in Figure 11, the overall decrease in accuracy for all features was relatively small (no higher than 0.04). This indicates that no single feature alone is responsible for the score obtained. This is to be expected since the features represent a distribution. For example, if one feature, such as deletion size 15, is randomly shuffled, the model should still be able to make a prediction using the other features of that group.

When focusing on different groups, small deletion sizes ranging from 2 to 6 had the most impact on the score when examining individual features. Other features, such as medium to large deletion sizes, insertions, and homology lengths, reported an average decrease in accuracy lower than 0.01, with some outliers. Showing that small deletions are the most discriminative features.

Following these findings, we have identified DS-Ins-HL (deletion size, insertion and homology length frequencies) as our selected frequency group. In addition, we found that small mutation sizes/lengths play an important role in predicting the genotype.

3.2. Target Sequence Representation

This leads us to now look at the second research question: "Can sequence representations improve the prediction performance of NHEJ repair deficiency?" Since the mutational spectra are target sequence dependent, we hypothesise that the model can use sequence information to find interactions between the mutational spectra features.

Initially, we discuss the impact of sequence encodings used alongside the frequency groups. Then, we examine if sequence embeddings can capture global sequence information, possibly further improving the prediction score obtained using only the mutational spectra.

3.2.1. Encoding Representations

In Figure 12, we compare the cross-validation predictive performances of the frequency distribution group (DS-Ins-HL) to other models where we have appended the selected sequence encodings as features (OHE & 1MF). Similar to Section 3.1, RF vastly outperforms LR for all the encoding techniques.

Starting with the RF model, we see significant changes in two encoding techniques. The 1-mer frequency (1MF) technique significantly improved from the original distribution ($94.44\% \pm 1.39\% \rightarrow 94.80\% \pm 1.43\%, p = 0.011$). Similarly, when the one-hot encoding technique features were combined with the 1-mer frequency (OHE-1MF), the model reported a significant improvement ($94.44\% \pm 1.39\% \rightarrow 95.07\% \pm 1.04\%, p = 0.031$). This improvement is likely derived from the 1-mer frequency features, as there was no statistical difference between the 1-mer frequency performance and the results obtained using OHE-1MF ($p = 0.436$).

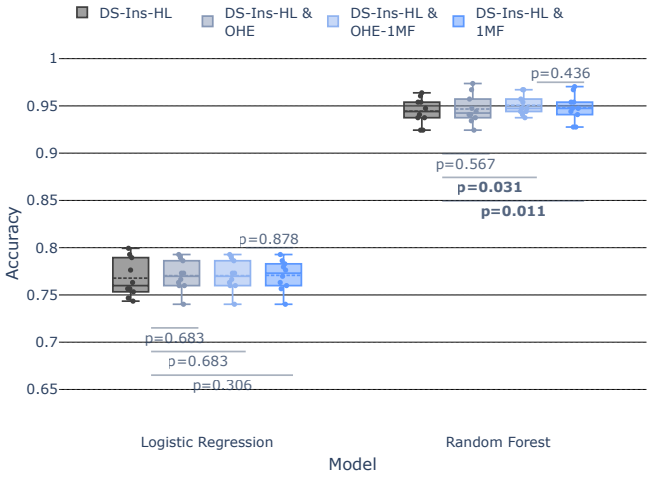


Fig. 12: Validation accuracy scores for the DS-Ins-HL feature group and all the encoding techniques (`url`). OHE—one-hot encoding; 1MF—1-mer frequency; OHE-1MF—OHE with 1MF.

In Appendix E.3, we elaborate on the performance of OHE-1MF. The evaluation of OHE-1MF showed that the OHE features push specific nucleotides to be correlated to the outcome class. This suggests that the 1MF encodings can capture similar information in a more compact representation.

In contrast to the RF findings, LR observed no significant changes with any encoding technique ($OHE : p = 0.683; OHE - 1MF : p = 0.683; 1MF : p = 0.306$). This indicates that the linear model could not extract additional information from the encoding representation. Based on these discoveries, we focus on 1-mer frequencies. 1-mer frequencies provide a simple representation of the target sequences with only four features. These encodings captured global information that was beneficial when using a non-linear algorithm.

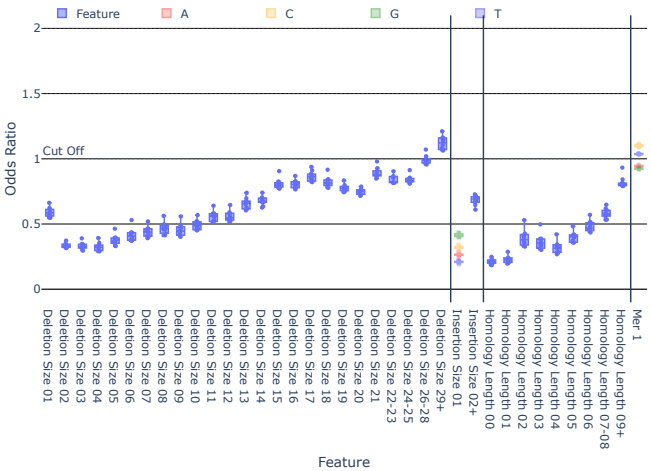


Fig. 13: The odds ratio values for DS-Ins-HL & 1MF (`url`). The data points are calculated from the models' coefficients created during cross-validation.

As shown in Figure 13, when reviewing the LR odds ratio for the 1-mer frequency, we observe similar trends to what was reported previously (discussed in Section 3.1.2). Small deletion sizes, insertions and homology lengths are still the most discriminative features. Focusing on the 1-mer frequency features, we see that if a target sequence has a higher A or G frequency on the forward strand, it is more likely that the NHEJ repair pathway will be utilised. Apart from this, we can note that C's and G's are more discriminative of the class than A's and T's due to the difference in the odds ratio value.

When reviewing the FPI for the 1-mer frequencies, as shown in Figure 14, we observed similar trends as those reported in Section 3.1.2. Most mutational features did not change from what we originally observed. In the 1-mer frequency features, only A-mer frequency slightly impacted the score (averaging at 0.004), comparable to medium-sized deletions (size 10). In contrast, other 1-mer frequencies had little to no impact on the score, similar to large deletions (averaging around 0.001).

These findings show that even with a simple representation technique, we can capture information beneficial for the prediction model, significantly improving the performance score of a non-linear model. Using 1-mer frequencies, made up of only four features, the RF discovered meaningful interaction between the mutational frequency features and the sequence encoding.

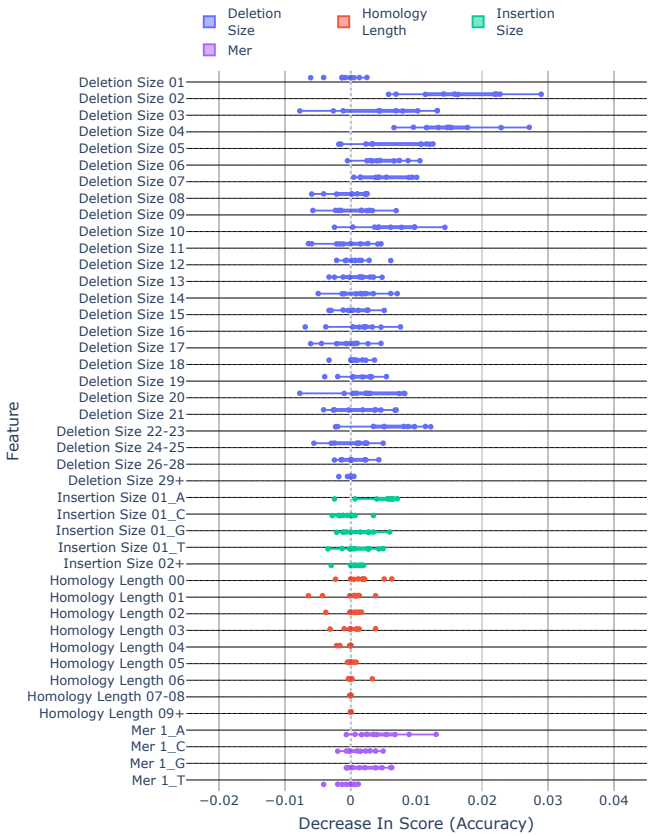


Fig. 14: The feature permutation importance on the validation set for DS-Ins-HL & 1MF (`url`). The value is averaged over 20 random shuffles.

3.2.2. Embedding Representations

After evaluating the encoding techniques, we look into the embedding representations. Using DNABERT (DB3, using $k = 3$) representations, we improved the prediction score further, as shown in Figure 15. Mean pooling (MP) and horizontal stacking (HS) observed significant improvements using the RF. However, the vertical stacking (VS) produced no improvements from the original frequency distribution groups. We use UMAP as a dimensionality reduction technique for the HS and VS since it can capture non-linear transformations between dimensions, potentially allowing for intricate relationships to be represented. To review PCA, an alternate dimensionality reduction technique, please refer to Appendix F.1. We compare the resulting performance values against the original scores (DS-Ins-HL) and 1-mer frequency (DS-Ins-HL & 1MF) features to evaluate how the resulting model performs against the previous techniques.

When reviewing the performance scores achieved using RF, most techniques reported a significant improvement, except for vertical stacking. Starting with the MP technique, it outperformed all other representations, with a resultant accuracy score of $96.12\% \pm 1.67\%, p = 0.009$, approximately a 1.70% increase from the original frequency features. Continuing with the HS, with a resulting dimensionality of 768 for each sequence, we observe a significant improvement, resulting in an average accuracy score of $94.44\% \pm 1.39\% \rightarrow 95.39\% \pm 1.48\%, p = 0.006$. When considering both embedding techniques, the reported median values are similar (96.05% and 96.38%), and even though there is a statistical difference between the reported results of both embedding techniques ($p = 0.012$), HS provides more flexibility in describing the token embeddings. This stacking technique allows for the resulting dimensionality to be a definable variable, where, as shown in Appendix F.2, even with a resulting dimensionality of 50 per sequence embedding, the model retained similar performance scores. This flexibility and control over the resulting feature space makes it ideal for problems with limited samples. Due to the curse of dimensionality, more samples are needed to use a larger dimensional space. Therefore, observing a significant improvement, even with a smaller dimensional space, HS can be integrated with problems with limited samples.

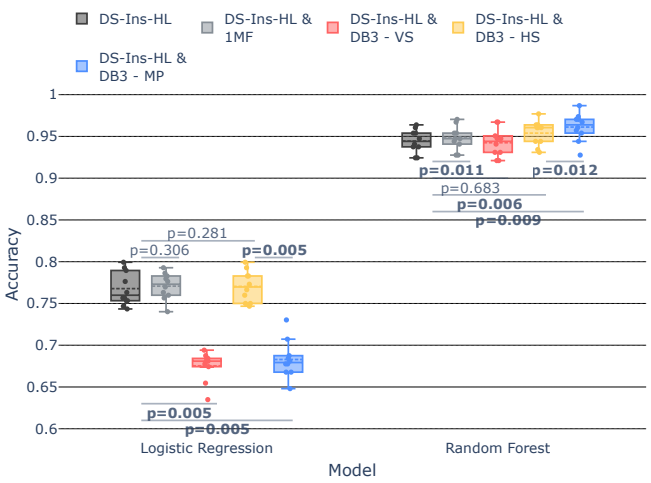


Fig. 15: Validation accuracy score of DS-Ins-HL, DS-Ins-HL & 1MF, DS-Ins-HL with vertical (VS), horizontal stacking (HS), and mean pooling (MP) on a pre-trained DNABERT model (DB3), with a $k = 3$ (`url`). HS and MP features have a dimensionality of 768. VS has a dimensionality of 795 (15 dimensions for each token). Both HS and VS are generated using UMAP as the dimensionality reduction technique.

Focusing on vertical stacking, VS is designed to allow interpretation of the token embeddings; however, the gained interpretability hinders the prediction power of both models. This suggests that during the stacking procedure, critical information is not retained. The tokens are treated as independent during the transformation. Therefore, we hypothesise that the dimensionality reduction is losing the necessary higher-order information described in these tokens. Even though the transformed space retained each token's positional information, this information is not descriptive enough of the neighbouring tokens and potential higher-order relationships. In addition, all tokens are treated as equal. If certain tokens or positions are more important than others, this stacking technique might not represent this since all will encode the same dimensionality.

Looking at LR, the model's baseline performance was not improved using any embedding technique. On the contrary, the score dropped ($76.78\% \pm 2.02\% \rightarrow 67.53\% \pm 1.76\%$) when using VS or MP ($76.78\% \pm 2.02\% \rightarrow 68.29\% \pm 2.26\%$).

Following the performance scores, the original trends persisted when reviewing the odds ratios and FPI. This was observed for both the MP (Appendix E.4) and HS (Appendix E.5), with a dimensionality of 768 per sequence. The frequency distribution features remained unmodified when evaluating the odds ratios for the LR. Small deletion sizes, insertions and homology lengths are still the most discriminative features. In contrast, newly introduced representation features resulted in noisy odds ratios. The LR model could not find linear relationships to generalise the embeddings (HS or MP). This further strengthens our belief that linear models like LR cannot capture critical information represented in a non-linear space. These noisy odds ratios also highlight why the LR observed a drop in performance.

When reviewing the FPI of the transformed DNABERT representations, the decrease for the original features remained relatively similar to the FPI values reported by DS-Ins-HL. However, for the representation features, in most cases, the decrease in score was close to zero, with most features reporting a decrease in score of $\approx 1 \times 10^{-6}$. This is likely because, just by shuffling one variable from 768, the impact on the resulting score is negligible. Suggesting that even though no single dimension is responsible for the improvement observed (with HS or MP), they collectively represent important information in the sequence.

From the observable significant improvements made when using both encoding and embedding representations, we have shown that sequence representations can improve the prediction performance of NHEJ repair deficiency. These results show that the Logistic Regression saw no improvement with encodings or embeddings. This shows that this linear model struggled to find meaningful representations between the target sequence and the mutational spectra. In contrast, the Random Forest significantly improved when introducing encodings or embeddings with the distribution frequencies. Based on these findings, prediction problems where mutational data is used for a prediction problem could always benefit by introducing 1-mer frequencies of the target sequences into their feature space. Similarly, when using non-linear algorithms, such as neural networks, instead of using 1-mer frequencies, one could opt to use higher-order representations.

Even without fine-tuning DNABERT and using the pre-trained representations, we have reported significant improvements when using the token embeddings, reduced through mean pooling. This suggests that the embeddings can be further refined through fine-tuning or domain adaptation, potentially yielding room for further improvements.

4. Conclusion

In this study, we demonstrated how representations of DNA sequence and mutational spectra can predict NHEJ repair deficient genotype. To tackle this problem, we first evaluated different mutation frequency groups comprising different mutation categories. Following this, we included target sequence representations with these mutation frequency groups.

We found that deletion sizes and homology length frequency distributions were the most predictive of the genotypes, with small deletion sizes being the most discriminative for this prediction problem. In addition, we showed that incorporating target sequence representations can further improve the prediction power of a model. Using Random Forests, even simple representations, such as 1-mer frequency encodings, achieve high accuracy ($94.80\% \pm 1.43\%$). With the use of more complex representations, such as embeddings, we were able to obtain even higher accuracy scores ($96.12\% \pm 1.67\%$). Even though there is little difference between the scores obtained, we believe both can be utilised in different scenarios. We advise using 1-mer frequencies when the amount of samples is limited. 1-mer frequency only requires four features to represent the entire sequence, irrespective of length. This allows it to describe generic information about the sequence without risking the curse of dimensionality. In contrast, DNABERT embeddings provide more contextual information about the k-mers and their neighbours at the cost of higher dimensional space. Therefore, such an approach might be better suited when a problem has many samples.

There are some limitations that need to be considered for future work. Firstly, the DNABERT model is not fine-tuned on the sequences we use. Fine-tuning allows token embeddings to be optimised on the input sequences and their labels. Since the labels are representative of the mutations, not the sequences, the prediction problem does not align with the fine-tuning confines of their architecture. For this prediction problem, we are limited to only the readily available pre-trained models. This restriction means that the embeddings might not perfectly represent the sequences. Secondly, the embeddings are not optimised for the sequences. We could obtain more representative embeddings through a domain adaptation function. Domain adaptation optimises a pre-trained model using previously unseen unlabelled data, whereas, during fine-tuning, the model is optimised based on the labels. When reviewing literature for domain adaptation on other BERT models, the number of samples commonly ranged from 100,000 to 1,000,000, whereas we only had 1,900 unlabelled target sequences. Finally, with the current technological capabilities, a single pathway from the cell cannot be removed, but specific genes can be knocked out to inhibit its ability to function. This means that even though the essential genes of a pathway have been knocked out of the cell, we cannot know with certainty that this pathway is not used or which pathway was used since there is no ground truth.

To improve on this work, we suggest exploring the domain adaptation of DNABERT to improve the already learned weights of the model without the need for labels. More meaningful and representative embeddings can be generated by applying a domain adaptation to the pre-trained DNABERT to better describe the new unseen data. Another potential improvement is to look into interpretable models, such as autoencoders, to gain insight into which k-mer is most impactful towards the prediction problem. Autoencoders aim to compress the representation into a smaller dimension and then reconstruct it with minimal loss of information. This type of reconstruction could allow for easier interpretability of the sequence k-mers, as opposed to the mean pooling or horizontal stacking. With mean pooling or horizontal stacking, we cannot know the impact of individual k-mer since the positional information is not preserved.

To conclude, we showed how target sequence representations can improve the prediction of Wild Type and NHEJ repair deficient genotypes. Based on our findings, prediction problems where mutational data is used could always benefit by introducing sequence representations. We see that 1-mer frequencies only need four features to describe the overall nucleotide composition of the sequence. Even though no gain or loss was reported with a linear model, models which can capture non-linear relationships observed significant improvements. When looking into more complex representations, such as DNABERT embeddings, we advise that these are used solely with models which can capture non-linear relationships. Even though our work uses embeddings from a pre-trained model, we believe future research could benefit by applying a domain adaptation function to the pre-trained model to make the representations potentially more descriptive. We aspire that such findings encourage future research to use representations, encodings or embeddings, with their mutational data to achieve more robust solutions.

References

1. J. Ross Chapman, Martin R. G. Taylor, and Simon J. Boulton. Playing the end game: Dna double-strand break repair pathway choice. *Molecular cell*, 47(4):497–510, 2012.
2. Ludovic Deriano and David B. Roth. Modernizing the nonhomologous end-joining repertoire: Alternative and classical nhej share the stage. *Annual review of genetics*, 47(1):433–455, 2013.
3. B. Pardo, B. Gómez-González, and A. Aguilera. Dna repair in mammalian cells. *Cellular and Molecular Life Sciences*, 66(6):1039–1056, Mar 2009.
4. Stephen P. Jackson and Jiri Bartek. The dna-damage response in human biology and disease. *Nature*, 461(7267):1071–1078, 2009.
5. Xin Zhao, Chengwen Wei, Jingjing Li, Poyuan Xing, Jingyao Li, Sihao Zheng, and Xuefeng Chen. Cell cycle-dependent control of homologous recombination. *Acta Biochimica et Biophysica Sinica*, 49(8):655–668, 05 2017.
6. Max W. Shen, Mandana Arbab, Jonathan Y. Hsu, Daniel Worstell, Sannie J. Culbertson, Olga Krabbe, Christopher A. Cassa, David R. Liu, David K. Gifford, and Richard I. Sherwood. Predictable and precise template-free crispr editing of pathogenic variants. *Nature*, 563(7733):646–651, 2018.
7. Helen Davies, Dominik Glodzik, Sandro Morganella, Lucy R Yates, Johan Staaf, Xueqing Zou, Manasa Ramakrishna, Sancha Martin, Sandrine Boyault, Anieta M Sieuwerts, Peter T Simpson, Tari A King, Keiran Raine, Jorunn E Eyfjord, Gu Kong, Åke Borg, Ewan Birney, Hendrik G

Stunnenberg, Marc J van de Vijver, Anne-Lise Børresen-Dale, John W M Martens, Paul N Span, Sunil R Lakhani, Anne Vincent-Salomon, Christos Sotiriou, Andrew Tutt, Alastair M Thompson, Steven Van Laere, Andrea L Richardson, Alain Viari, Peter J Campbell, Michael R Stratton, and Serena Nik-Zainal. HRDetect is a predictor of BRCA1 and BRCA2 deficiency based on mutational signatures. *Nature Medicine*, 23(4):517–525, March 2017.

8. F Ann Ran, Patrick D Hsu, Jason Wright, Vineeta Agarwala, David A Scott, and Feng Zhang. Genome engineering using the CRISPR-cas9 system. *Nature Protocols*, 8(11):2281–2308, October 2013.
9. Megan van Overbeek, Daniel Capurso, Matthew M Carter, Matthew S Thompson, Elizabeth Frias, Carsten Russ, John S Reece-Hoyes, Christopher Nye, Scott Gradia, Bastien Vidal, et al. Dna repair profiling reveals nonrandom outcomes at cas9-mediated breaks. *Molecular cell*, 63(4):633–646, 2016.
10. M. Jasin and R. Rothstein. Repair of strand breaks by homologous recombination. *Cold Spring Harbor Perspectives in Biology*, 5(11):a012740–a012740, October 2013.
11. Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding, 2019.
12. Hitoshi Iuchi, Taro Matsutani, Keisuke Yamada, Natsuki Iwano, Shunsuke Sumi, Shion Hosoda, Shitao Zhao, Tsukasa Fukunaga, and Michiaki Hamada. Representation learning applications in biological sequence analysis. *Comput Struct Biotechnol J*, 19:3198–3208, May 2021.
13. Yanrong Ji, Zhihan Zhou, Han Liu, and Ramana V Davuluri. DNABERT: pre-trained Bidirectional Encoder Representations from Transformers model for DNA-language in genome. *Bioinformatics*, 37(15):2112–2120, 02 2021.
14. Misganaw Asmamaw and Belay Zawdie. Mechanism and applications of CRISPR/Cas-9-Mediated genome editing. *Biologics*, 15:353–361, August 2021.
15. Madeleine Oman, Aqsa Alam, and Rob W. Ness. How sequence context-dependent mutability drives mutation rate variation in the genome. *Genome biology and evolution*, 14(3), 2022.
16. T. Anthony Sun and Peter A. Lind. Distribution of mutation rates challenges evolutionary predictability. *Microbiology*, 169(5), 2023.
17. Fixed-width bins. `https://www.ibm.com/docs/en/spss-modeler/18.2.2?topic=node-fixed-width-bins`, 2021. Accessed: 2023-08-31.
18. Machine learning glossary. `https://developers.google.com/machine-learning/glossary#quantile-bucketing`. Accessed: 2023-08-31.
19. Kutubuddin A. Molla, Justin Shih, Matthew S. Wheatley, and Yinong Yang. Predictable NHEJ insertion and assessment of HDR editing strategies in plants. *Frontiers in Genome Editing*, 4, March 2022.
20. Pau Rodríguez, Miguel A. Bautista, Jordi Gonzàlez, and Sergio Escalera. Beyond one-hot encoding: Lower dimensional target embedding. *Image and Vision Computing*, 75:21–31, 2018.
21. Rishi Bommasani, Kelly Davis, and Claire Cardie. Interpreting Pretrained Contextualized Representations via Reductions to Static Embeddings. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 4758–4781, Online, July 2020. Association for Computational Linguistics.
22. Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need, 2023.
23. Richard Bellman. Dynamic programming. *Science*, 153(3731):34–37, 1966.
24. Bram Vanroy. Generate raw word embeddings using transformer models like bert for downstream process. `https://discuss.huggingface.co/t/generate-raw-word-embeddings-using-transformer-models-like-bert-for-downstream-process/2958/2`, 2020. Accessed: 2023-08-31.
25. Ian T Jolliffe and Jorge Cadima. Principal component analysis: a review and recent developments. *Philos Trans A Math Phys Eng Sci*, 374(2065):20150202, April 2016.
26. Tim Sainburg, Leland McInnes, and Timothy Q Gentner. Parametric umap embeddings for representation and semisupervised learning. *Neural Computation*, 33(11):2881–2907, 2021.
27. Emanuele Vivoli. Reduce the number of features of bert embeddings. `https://discuss.huggingface.co/t/reduce-the-number-of-features-of-bert-embeddings/3424/2`, 2021. Accessed: 2023-08-31.
28. Leo Breiman. Random forests. *Machine Learning*, 45(1):5–32, Oct 2001.
29. Lars Buitinck, Gilles Louppe, Mathieu Blondel, Fabian Pedregosa, Andreas Mueller, Olivier Grisel, Vlad Niculae, Peter Prettenhofer, Alexandre Gramfort, Jaques Grobler, Robert Layton, Jake VanderPlas, Arnaud Joly, Brian Holt, and Gaël Varoquaux. API design for machine learning software: experiences from the scikit-learn project. In *ECML PKDD Workshop: Languages for Data Mining and Machine Learning*, pages 108–122, 2013.
30. Magdalena Szumilas. Explaining odds ratios. *Journal of the Canadian Academy of Child and Adolescent Psychiatry*, 19(3):227, 2010.
31. Permutation feature importance. `https://scikit-learn.org/stable/modules/permutation_importance.html`. Accessed: 2023-08-31.
32. Plotly Technologies Inc. Collaborative data science. `https://plot.ly`, 2015. Accessed: 2023-08-31.

Supplementary Material

A. Data

In this appendix, we discuss the data in more detail. The focus is on the target sequence nucleotide composition and the rare mutations such as deletion size 40. After this, we discuss handling the ambiguities found in the data during the pre-preprocessing.

A.1. Data Description

The data consists of target sequences and their respective mutations under two genotypes, Wild Type and NHEJ repair Deficient. Shen et al. [1] generated 2,000 target sequences, each consisting of 55 nucleotides (A/C/G/T). As shown in Figure 16, the position reported is relative to the cut site. This means that the range starts at -27 (27 nucleotides left of the cut site) and ends at 27 (27 nucleotides right of the cut site).

These target sequences were designed by determining the distribution of four statistics in sequences from the human genome. These four statistics are GC content, the total sum of bases

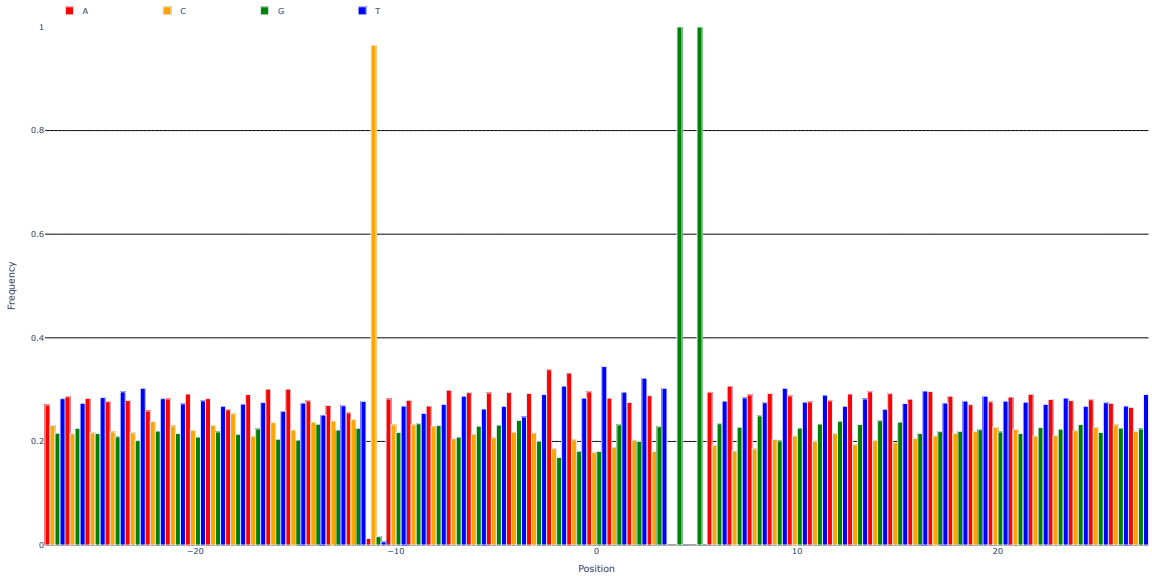


Fig. 16: Frequency of each nucleotide at each position. The PAM is from position 3 to position 5 (NGG). Positions range from -27 to 27. A's and T's appear to be more frequent than C's and G's. At position -11, we see an outlier, with 96% of all target sequences having C.

participating in microhomology for 3-27-bp deletions, Azimuth predicted on-target efficiency score, and the statistical entropy of the predicted 3-27-bp deletion length distribution from a previous version of inDelphi [1]. These sequences are then randomly generated using a random search on the different quintiles of the four human genome statistics.

When evaluating these sequences, we discovered slight abnormalities that might impact the encoding representations. As shown in Figure 16, most positions report similar nucleotide frequencies, ranging between 20% and 30% on average. Focusing on the cut site and its neighbouring nucleotides, Figure 16 shows that A's and T's are more frequent than C's and G's. As expected, the PAM sequence between positions +3 and +5 shows that only GG are reported at positions +4 and +5. This is because the PAM sequence consists of the NGG sequence, where N (position +3) is used as a placeholder for any nucleotide, and GG is fixed throughout all of the sequences. However, at position -11, 96% of the target sequences observe the nucleotide C at this specific location, with T being the least observable, with only 0.63%. This outlier could be one of the reasons why, as discussed in Appendix E.2, the odds ratio reports T at this position as having a high association with the NHEJ repair deficient class.

Besides target sequences, mutations are a key aspect of the study. As discussed in Section 2.1, we use mutations from two cell populations, one with the Wild Type genotype and the other with the NHEJ repair deficient genotype. The NHEJ repair deficient genotype has two essential genes, the PRKDC and LIG4 genes, knocked out, attempting to impede the use of the NHEJ repair pathway. The PRKDC (DNA-PKcs) gene encodes a protein that forms a complex with other proteins, including Ku70 and Ku80, to recognise and bind to the broken ends of DNA. This complex, called DNA-PK, then recruits other repair factors to the site of damage, such as LIG4 (DNA ligase IV), which seals the broken ends of DNA together [2]. The LIG4 gene encodes the DNA ligase IV protein, which is responsible for ligating the broken DNA ends together during NHEJ repair [3]. By hindering these genes, believed to be accountable for recognising and ligating the DSB ends, the NHEJ repair pathway should not be active in this cell population.

When looking at the mutational data provided in Table 3, we can see a description alongside the range each attribute covers. The homology length and the type attributes were unaffected by the filtering process. However, size and start saw a drop in the distinct mutations they observed, as shown in the table.

Following the data pre-processing, rare/uncommon mutations still occur. In Figure 17, large deletion sizes (30 onwards) are rarely observed in most sequences. For both classes, these types of

Table 3. The data attributes, a description and their values/ranges.

Attribute	Description	Pre-Filter	Post-Filter
Type	Type of mutation	Deletion/ Insertion	Deletion/ Insertion
Size	Size of mutation	Del 1 - 44 Ins 1 - 22	Del 1 - 40 Ins 1 - 18
Start	Start position of mutation	Del -31 - 17 Ins -15 - 18	Del -31 - 5 Ins 0
Ins. Seq.	Inserted sequence	[A|C|G|T] {1-22}	[A|C|G|T] {1-18}
MH Length	Microhomology Length	0 - 12	0 - 12
Count Events	Times mutation occurred	2 - 2115601	2 - 2072131

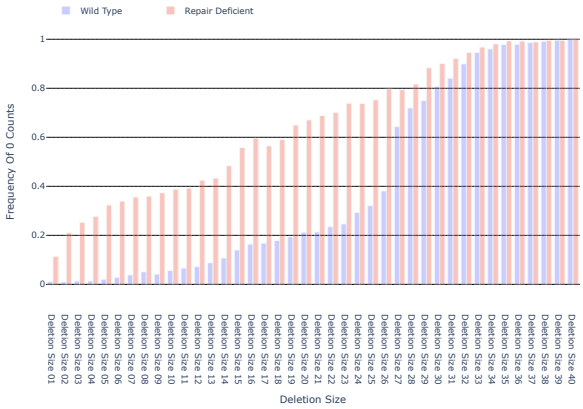


Fig. 17: Zero counts for all the deletion size mutations. The NHEJ Repair deficient class has a higher rate of zero counts in all the features. Any deletion equal to or larger than 30, had a zero count greater than 80% for both classes.

deletion sizes report that more than 80% of the sequences never observe such mutations.

This problem of never observing a mutation (zero counts) was more prominent with insertions, as shown in Figure 18. Only insertion sizes 1 and 2 had more than 50% of the target sequences report such insertion sizes. Following a similar trend to the deletion sizes, the NHEJ repair deficient class had a higher rate of zero counts, with 80% of the target sequences not reporting an insertion of size 3 (52% for the Wild Type class).

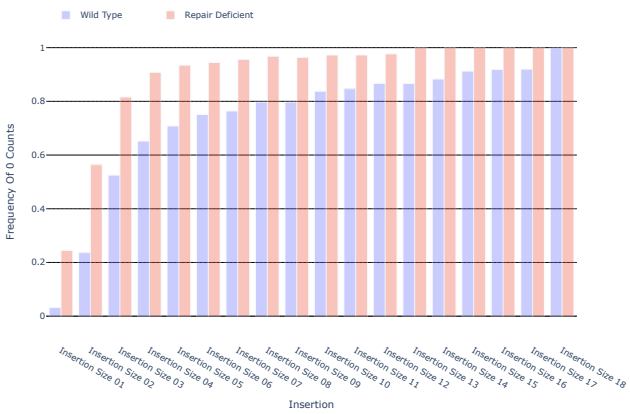


Fig. 18: Zero counts for all the insertion size mutations. The NHEJ Repair deficient class has a higher rate of zero counts in all the features. Any deletion equal to or larger than 4 had a zero count greater than 60% for both classes.

A.2. Data Ambiguities

During the mutational outcome generation process, ambiguities impact the reported attribute values. As previously discussed, the mutation data attributes provided are calculated by aligning the original target sequence and the repair product to the left and observing all the differences. The workflow reads from left to right and reports the first difference observed. When working with DNA sequence data, ambiguities arise due to repeats. Repeats in the sequence cause uncertainties in the reported attribute values since there is no way of knowing the correct start.

In Figure 19, we see ambiguous MH deletion and Insertion cases. Starting with the MH Deletion, we cannot know whether the deletion was at position +01 (the value reported) or at -01 since the resulting repair product would be the same for all options highlighted. Similarly, in the insertion case, if an "A" nucleotide is inserted at position -02 or +01, the repair product would be equivalent in both cases.

There are no means of confirming the exact start of such cases. Therefore, we opted to include ambiguous cases in our study to avoid losing potentially important information and maintain a variety of mutations. To identify ambiguous mutations that we retain, we use a number of steps to check their validity. Initially, we start by identifying normal from ambiguous cases and checking for repeats. Afterwards, we identify the ambiguities that match the filtering condition (occur at or span the cut site) for each ambiguous mutation.

Specifically, in the case of insertion, we check if repeats (partial, that is, a part of the inserted sequence or complete, the entire inserted sequence) occur. We then check the range of the repeats, and if any of the possible cases yields the same repair product and satisfies the filtering condition, we set the start to the original start to the valid start. For example, in Figure 19, the Insertion start would be updated from +02 to 00 since this satisfies the filtering condition.

Similarly, in the case of deletions, we check if repeats occur. We then check the range of the repeats, and if any of the possible cases yield the same repair product and satisfy the filtering condition, we set the start to the furthest possible valid start. For example, in Figure 19, the MH deletion start would be updated from +01 to -01.

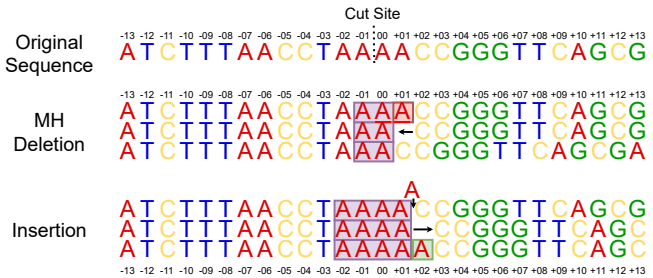


Fig. 19: Two examples of ambiguous mutations. In the first example, the MH deletion, the nucleotide "A" was deleted, with a microhomology length of one. As highlighted in purple, the workflow cannot be certain which nucleotide was deleted, since all the highlighted nucleotides result in the same repair product. Similarly, the Insertion example shows the nucleotide "A" inserted at position 2. All neighbouring nucleotides are repeats, thus we cannot know which nucleotide was inserted.

B. Data Filtering

The following appendix shows an accepted case and a filtered-out case of the different mutation types based on the filtering condition.

Starting with MH-Deletions, we see an accepted and a filtered case in Figure 20. The first set of nucleotides is accepted since the deleted nucleotides span the cut site. The second sequence set is filtered out since the mutations occurred before the cut site.

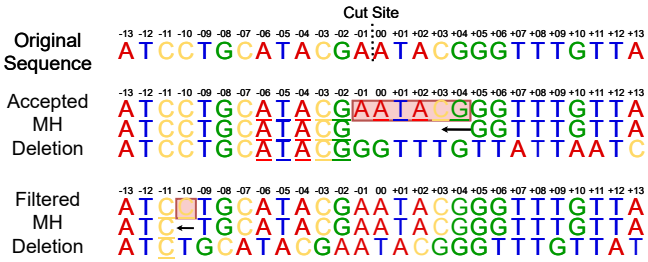


Fig. 20: An accepted and a filtered-out microhomology deletions. The first row shows a part of the unmodified target sequence. In the first example, the accepted MH deletion, the sequence "AATACG" starting at location -1 was deleted, causing the nucleotides on the right to shift towards the other nucleotides. In the second example, the MH deletion, the sequence "C" was deleted. However, this deletion did not meet our condition.

Figure 21 shows an accepted and filtered case for MH-Less Deletions. The first sequence is retained since the deleted nucleotides span the cut site. The second sequence is filtered out since the mutation falls right before the cut site.

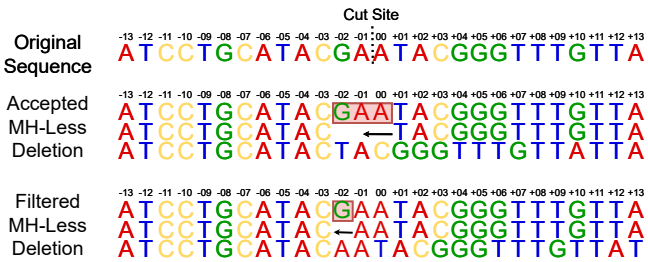


Fig. 21: An accepted and filtered-out microhomology-less deletion. The first row shows a part of the unmodified target sequence. In the first example, the accepted MH-Less deletion, the sequence "GAA" starting at location -2 was deleted, causing the nucleotides on the right to shift towards the other nucleotides. In the second example, the filtered MH-Less deletion, the sequence "G" was deleted. However, this deletion did not meet our condition.

Figure 22 shows an accepted and a filtered-out case for Insertions. The first sequence is retained since the inserted nucleotides occur at the cut site. The second sequence is filtered out since it falls before the cut site.

Post filtering, we compare the original count events observed and the resulting count events. This comparison allows us to understand the overall impact of the filtering on the mutational

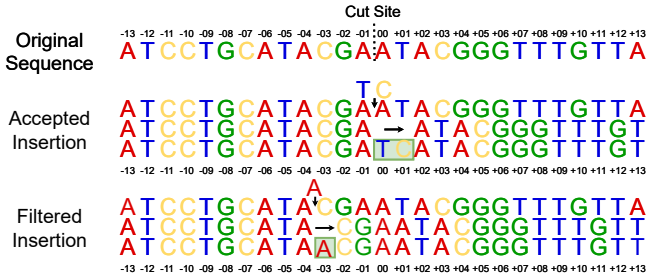


Fig. 22: An accepted and filtered-out insertion. The first row shows a part of the unmodified target sequence. In the first example, the accepted insertion, the sequence "TC" starting at location 0 was inserted, causing the nucleotides on the right to shift. In the second example, the filtered insertion, the nucleotide "A" was inserted at -03. However, this insertion did not meet our condition.

data. In Table 4, we can see the original count events and the count event after filtering. Overall, we observe a loss of 19% from the original count events, with most mutations lost being insertions, observing a total loss of 40% of all count events.

Table 4. Count events for different types of mutations. Describing the Pre-filtering and post-filtering totals, alongside the observable change in their values.

	Pre-Filtering	Post-Filtering	Observable Change
Insertion	122,131,241	72,885,203 (N: 8,438,182 A: 64,447,021)	-49,246,038 (40.322% lost)
MH-Less Deletions	147,344,730	98,786,714	-48,558,016 (32.955% lost)
MH Deletions	568,690,659	507,142,955 (N: 477,870,558, A: 29,272,397)	-61,547,704 (10.823% lost)
Total	838,166,630	678,814,872	-159,351,758 (19.012% lost)

N represents the normal insertions/deletions

A represents the ambiguous insertions/deletions

Percentage Loss is calculated: $loss = \frac{original - resulting}{|original|} \times 100$

C. Mutational Outcome Featurization

In this appendix, we show the features which make the distribution groups. Table 5 shows the standard, unmodified frequency distribution features. In Table 6, we see the features after applying a binning of 1%.

When a distribution group has more than one set of features, for example, DS-Ins-HL, these distribution groups use all the features in each group. This means that in the case of DS-Ins-HL, this group will use the Deletion size frequency distribution, the Insertion frequency distribution and the Homology length frequency distribution. It is important to note that when homology length and insertion frequency distributions are used together, the Insertion feature in Homology length frequency is removed (one

less feature in this set) since it is described better using Insertion frequency distribution. When employing a binning technique, the percentile is calculated for each group. The resulting features are then used instead of the original features. For example, DS-Ins-HL results in 57 features for the unbinned group and 39 for the binned group. This is further explained in Appendix D.

Table 5. The frequency features each experiment before binning to train and predict the respective models.

	# of Features	Feature Names
Mutation Type Frequency	3	Insertion, MH-Deletion, MH-less Deletion
HL[1] Frequency Distribution	13	Insertion, No MH, HL from 01 to 12 (excluding 10)
Ins[2] Frequency Distribution	5	Ins (A), Ins (C), Ins (G), Ins (T), Insertion 2+
DS[3] Frequency Distribution	40	Deletion Size from 01 to 40
MH DS[4] Frequency Distribution	40	Deletion Size from 01 to 40
MH-Less DS[5] Frequency Distribution	39	Deletion Size from 01 to 39

[1]Homology Length; [2] Insertion;
[3]Deletion Size; [4] Microhomology Deletion Size;
[5]Microhomology-Less Deletion Size.

Table 6. The frequency features of each experiment after binning with a 1% used to train and predict the respective models.

	# of Features	Feature Names
Mutation Type Frequency	3	Insertion, MH-Deletion, MH-less Deletion
HL[1] Frequency Distribution	10	Insertion, No MH, HL from 01 to 06, 07-08, 9+
Ins[2] Frequency Distribution	5	Ins (A), Ins (C), Ins (G), Ins (T), Insertion 2+
DS[3] Frequency Distribution	25	DS from 01 to 21, 22-23, 24-25, 26-28, 29+
MH DS[4] Frequency Distribution	25	MH DS from 01 to 21, 22-23, 24-25, 26-28, 29+
MH-Less DS[5] Frequency Distribution	25	MH-Less DS from 01 to 18, 19-20, 21, 22-23, 24-25, 26-27, 28-30, 31+

"from x to y" denotes a single feature to represent each unique number starting from x and finishing with y.

The symbol "-" denotes that those numbers falling in that range (inclusive), are assigned that label.

The symbol "+" denotes that any number equal or larger than that is assigned that label.

D. Feature Binning

Data binning, also known as data discrete binning or data bucketing, is a data pre-processing technique used to reduce the effects of minor observation errors [4]. We used a percentile binning technique over a fixed-width binning technique (such as Sturges [5] or Doane [6]). Percentile binning allows us to discretise the features reporting high zero counts into a single feature while maintaining the other features unmodified.

In order to bin features, we use the following condition: identify a percentile value n% and the data group to discretise (discussed in Appendix D.1). For each feature in the group (ascending order), check if this feature has at least n% of the sample report a non-zero count. If this condition is satisfied, keep the feature and move to the next feature. Otherwise, combine the following m features until the binned features have n% of the samples report a non-zero count.

As shown in the next sections, binning yielded no significant improvement from the standard features. On the contrary, when using higher percentile values, such as n=7%, we saw a drop in the accuracy score (approximately 5%) for the deletion size frequency features.

D.1. Binning Groups & Resulting Features

Binning is applied to four groups of features: Deletion Size, Homology Length, MH Deletion Size, and MH-Less Deletion Size. These four groups reflect the attributes; however, we focus on the attributes that report high sparsity in their features. Even though MH and MH-Less Deletion Sizes are derived from the Deletion Size attribute, we still opted to treat them as independent groups since both groups describe different information, and the distribution of one group differs.

Table 7 shows the resulting features after each group's binning at a different percentile. As shown in Appendix D.2, the original accuracy (no binning) was not improved with any binning percentile. On the contrary, a performance drop is observed in certain groups as the percentile value increases. We compared the different bin values and combinations, such as deletion size with a percentile of 2% and homology length of 4%, against the standard features.

We opted for a 1% binning value because it can handle previously unseen samples, such as deletion size 50. In addition, this percentile value allowed for minimal loss in resolution.

Table 7. The resulting features of each attribute group at different bin percentile values, ranging from 1 to 10.

Bin Size	Deletion Size	Homology Length	MH Deletion Size	MH-Less Deletion Size
1%	DS from 01 to 21, 22-23, 24-25, 26-28, 29+	Insertion, No MH, HL from 01 to 06, 07-08, 9+	MH DS from 01 to 21, 22-23, 24-25, 26-28, 29+	MH-Less DS from 01 to 18, 19-20,21, 22-23, 24-25, 26-27, 28-30, 31+
2%	DS from 01 to 14, 15-16, 17-18,19-20, 21-22, 23-25, 26+	Insertion, No MH, HL from 01 to 06, 07+	MH DS from 01 to 14, 15-16, 17-18, 19-20, 21-22, 23-25, 26+	MH-Less DS from 01 to 13, 14-15, 16-17, 18-19, 20-21, 22-24, 25-28, 29+
3%	DS from 01 to 13, 14-15, 16-17, 18-19, 20-22, 23-27, 28+	Insertion, No MH, HL from 01 to 05, 06-07, 08+	MH DS from 01 to 13, 14-15, 16-17, 18-19, 20-22, 23-27, 28+	MH-Less DS from 01 to 09, 10-11, 12-13, 14-15, 16-18, 19-22, 23-26, 27+
4%	DS from 01 to 11, 12-13, 14-15, 16-18, 19-22, 23-31, 32+	Insertion, No MH, HL from 01 to 05, 06-08, 09+	MH DS from 01 to 12, 13-14, 15-17, 18-20, 21-25, 26+	MH-Less DS from 01 to 08, 09-10, 11-12, 13-15, 16-18, 19-23, 24+
5%	DS from 01 to 10, 11-12, 13-14, 15-18, 19-23, 24+	Insertion, No MH, HL from 01 to 05, 6+	MH DS from 01 to 11, 12-13, 14-16, 17-20, 21-27, 28+	MH-Less DS from 01 to 06, 07-08, 09-10, 11-12, 13-15, 16-20, 21-26, 27+
6%	DS from 01 to 06, 07-08, 09, 10-11, 12-13, 14-17, 18-22, 23+	Insertion, No MH, HL from 01 to 05, 06+	MH DS 01, 02-03, from 04 to 09, 10-11, 12-13, 14-16, 17-20, 21-31, 32+	MH-Less DS from 01 to 06, 07-08, 09-10, 11-13, 14-18, 19-25, 26+
7%	DS 01, 02-03, 04, 05, 06, 07-08, 09-10, 11-12, 13-15, 16-20, 21+	Insertion, No MH, HL from 01 to 04, 05-06, 07+	MH DS 01-02, from 03 to 06, 07-08, 09-10, 11-12, 13-15, 16-20, 21+	MH-Less DS from 01 to 05, 06-07, 08-09, 10-12, 13-17, 18-24, 25+
8%	DS 01, 02-03, 04, 05-06, 07-08, 09-10, 11-12, 13-16, 17-23, 24+	Insertion, No MH, HL from 01 to 04, 05-06, 07+	MH DS 01-02, 03-04, 05-06, 07-08, 09-10, 11-12, 13-15, 16-21, 22+	MH-Less DS from 01 to 04, 05-06, 07-08, 09-11, 12-16, 17-24, 25+
9%	DS 01-02, 03-04, 05-06, 07-08, 09-10, 11-13, 14-18, 19+	Insertion, No MH, HL from 01 to 04, 05-06, 07+	MH DS 01-02, 03-04, 05-06, 07-08, 09-10, 11-12, 13-16, 17-23, 24+	MH-Less DS 01, 02, 03-04, 05-06, 07-09, 10-13, 14-21, 22+
10%	DS 01-02, 03-04, 05-06, 07-08, 09-10, 11-13, 14-19, 20+	Insertion, No MH, HL from 01 to 04, 05-07, 08+	MH DS 01-02, 03-04, 05-06, 07-08, 09-10, 11-13, 14-19, 20+	MH-Less DS 01, 02, 03-04, 05-06, 07-09, 10-13, 14-21, 22+

"from x to y" denotes a single feature to represent each unique number starting from x and finishing with y (inclusive).

The symbol "-" denotes that those numbers falling in that range (inclusive), are assigned that label.

The symbol "+" denotes that any number equal or larger than that is assigned that label.

D.2. Bin Group Performance

This section compares the impact on the accuracy score when using ten different binning percentiles (point on the figure) against the standard frequency features (red shading and dotted line).

Starting with the homology length binning, we do not see any changes in the performance for either the LR or RF. This is because the variance in the figures is almost identical to the variance we achieve with the standard features, as shown in Figure 23 and Figure 24, respectively. NB: for homology length, some different binning sizes return identical features, for example, 5% and 6% or 7%, 8%, and 9%.

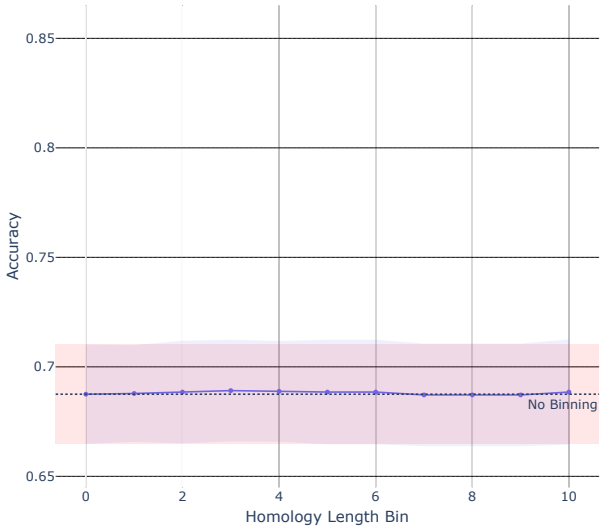


Fig. 23: Validation accuracy scores for all bin percentile values when using LR (`url`). There is no difference between the unbinned features and the binned features under any size.

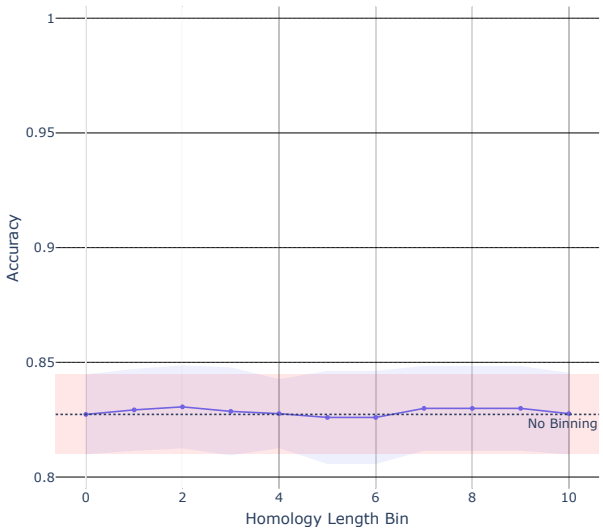


Fig. 24: Validation accuracy scores for all different bin percentile values when using RF (`url`). There is no difference between the unbinned features and the binned features under any size.

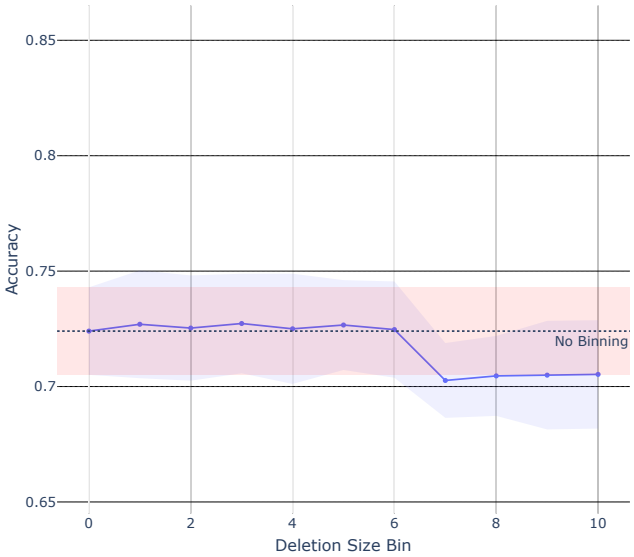


Fig. 25: Validation accuracy scores for all different bin percentile values when using LR. (`url`) There is a drop in performance when a bin size of 7% or more is used.

Looking at the deletion size binning, no improvements are observed under any bin size. On the contrary, as shown in Figure 25 and Figure 26 for the LR and RF models, we see a significant drop in performance when the bin size is $\geqslant 7$. In the case of the LR model, performance stabilises after the drop, whereas the RF model sees a further drop in performance. This suggests that the binning technique combines essential features. When small deletion sizes, smaller than 10, began to be binned together, this resulted in a drop. This further reinforces the understanding that small and medium size deletions are important for both models to discriminate between classes.

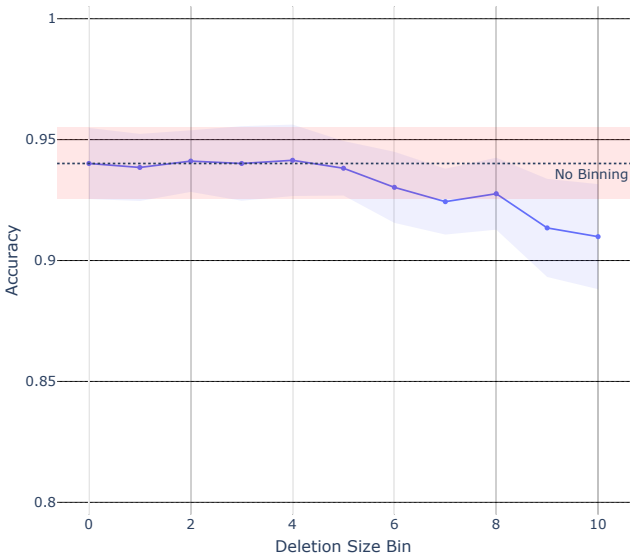


Fig. 26: Validation accuracy scores for all different bin percentile values when using RF (`url`). There is a drop in performance when a bin size of 6% or more is used.

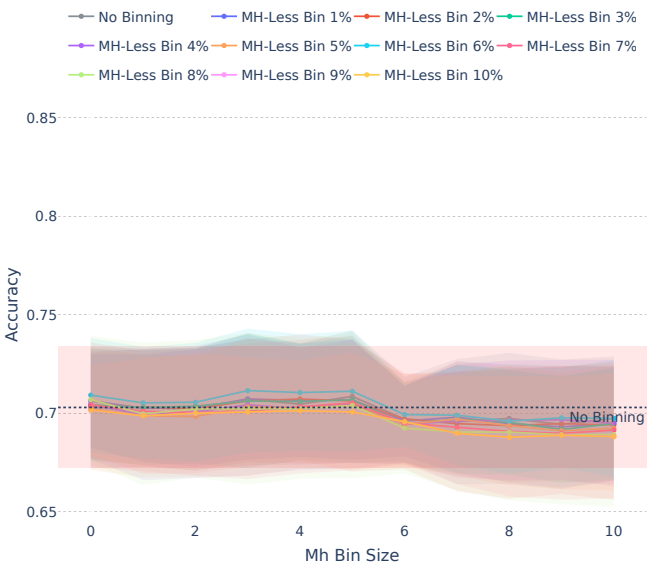


Fig. 27: Validation accuracy scores for all different bin percentile values when using LR (`url`). There is a drop in performance when a bin size of 6% or more is used, across all the different MH-Less bin sizes

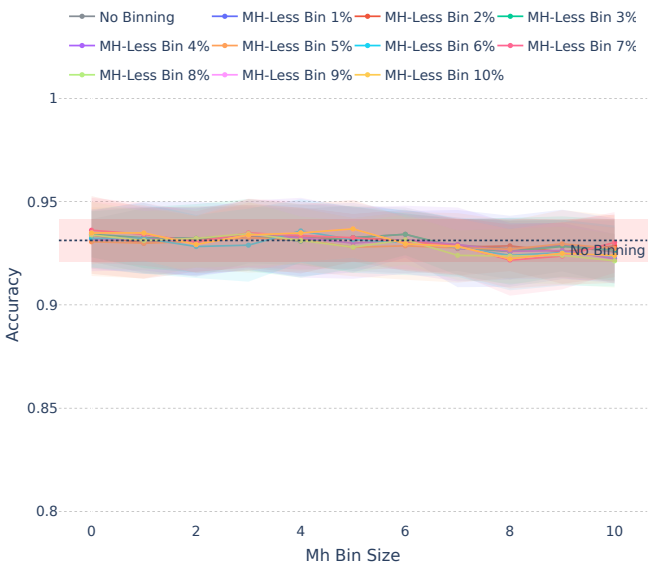


Fig. 28: Validation accuracy scores for all different bin percentile values when using RF (`url`). There is a drop in performance when a bin size of 6% or more is used, across all the different MH-Less bin sizes

Focusing on the MH and MH-Less deletion sizes, we see similar trends to what we saw in the deletion sizes. Figure 27 and Figure 28 show the performance obtained for the LR and RF, respectively. The x-axis is the MH deletion bin sizes, whereas the legends are the different combinations of the MH-Less deletion bin size. As we can see in both figures, a drop in accuracy is seen when the MH deletion sizes are binned with a percentile of 6% or higher. In contrast, the MH-Less (lines) seem stable throughout all the different percentile values. After using a bin size of 6% and onwards, the drop in accuracy is likely attributed to the MH binning deletion sizes 2 and 3 into a single group. Even though these values have high zero counts, the reduction in sparsity negatively impacted the overall performance, further confirming our understanding of short deletion sizes.

E. Additional Evaluations

This appendix discusses the additional encoding and embedding techniques reviewed in this study. Specifically, we look into 2-mer frequency encoding, one-hot encoding (with and without 1-mer frequencies), the DNABERT resulting odds ratios and feature permutation importance.

E.1. 2-Mer Frequency Encoding

In our research, we focus on 1-mer frequency (1MF) encoding. We use 1-mers instead of 2-mers (2MF) due to the statistical difference in score between 1MF and 2MF and the ease of interpretability.

Figure 29 shows there is no statistical difference between the original group distribution DS-Ins-HL and when 2MF frequency encodings are used alongside these features (LR: $p = 0.201$, RF: $p = 0.355$). When working with 1MF, we see a significant improvement from DS-Ins-HL under the RF model ($p = 0.011$). Comparing 1MF with 2MF, the accuracy scores reported no significant changes under either model (LR: $p = 0.835$, RF: $p = 0.303$). This indicates there is no added benefit of using 2MF instead of 1MF. On the contrary, we see a non-significant increase in the inner quartile of the box plot of the cross-validation scores on the RF model for 2MF.

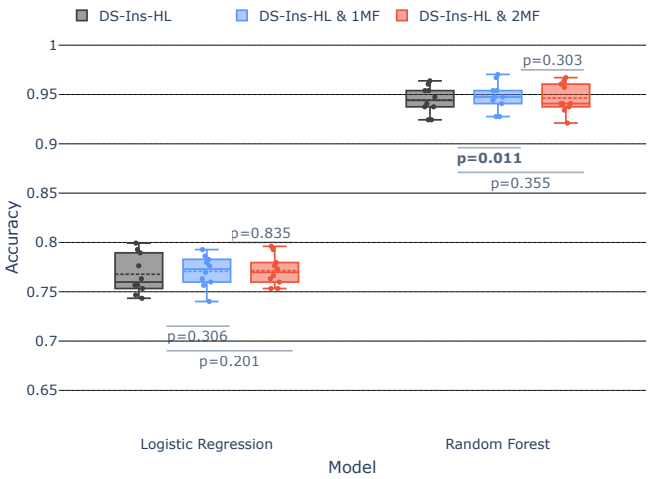


Fig. 29: Validation accuracy scores for the DS-Ins-HL feature group, DS-Ins-HL & 1MF, and DS-Ins-HL & 2MF (`url`). There is no statistical difference between 1MF and 2MF.

Looking at the Odds Ratios in Figure 30, we observe a similar trend to the 1-mer frequency encoding in Figure 13. Across all 16 2MF features, the positive and negative association to the NHEJ repair deficient class is split. On a closer look, the association depends on the ending nucleotide of the 2MF features. For example, all the 2-mers ending with C: AC, CC, GC, and TC, positively associate with the NHEJ repair deficient class. This is reflected along the other nucleotides, where A and G appear to have an overall negative association for all 2-mers, and C and T have a positive association. The interesting trend which emerges from these observations is that when referring back to the odds ratio of the 1MF (Figure 13), these followed the same trend we saw initially. The C's and T's have a positive association, and A's and G's have a negative association.

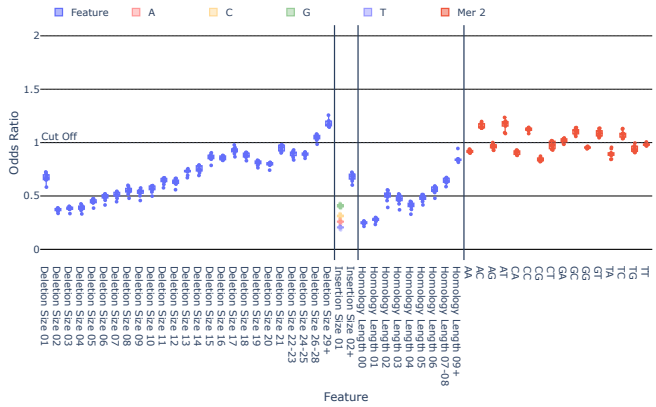


Fig. 30: The odds ratio values for DS-Ins-HL & 2MF. The 2-Mer frequency encoding features are shaded in bright red.

Finally, looking at the FPI for the RF model when using DS-Ins-HL & 2MF, we see that most of the 2MF features had little to no impact on the resulting score. The small deletion size features had the most impact, and the 2MF features had an average decrease in score of approximately 0.001.

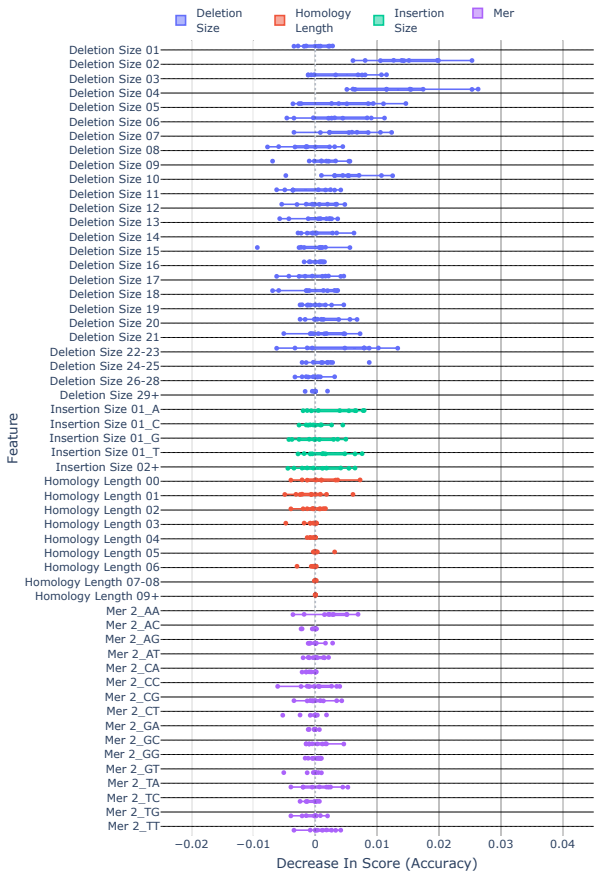


Fig. 31: The feature permutation importance on the validation set for DS-Ins-HL & 2MF. The x-axis represents the change in score compared to the original score, whereas the y-axis represents all of the features available in the experiment.

E.2. One-Hot Encoding

Apart from the k-mer frequency encoding, we also evaluated the one-hot encoding. Even though, as shown in Figure 12, there was no significant improvement when adding OHE to DS-Ins-HL (LR: $p = 0.683$, RF: $p = 0.567$), we look into the odds ratios and FPI to learn potential interesting trends these models are capturing.

Looking at the Odds Ratios in Figure 32, we observe that positional information contributes to the magnitude of the classes. Firstly, the odds ratio shows us that nucleotides positioned between -10 and +03 have a larger magnitude than nucleotides outside this area. This means that these positions are more discriminative of the class (ratio values range between 0.5 and 1.5), with the most discriminative feature being the nucleotide to the left of the cut site. Secondly, focusing on the individual nucleotides, C's and T's, similar to what we saw in 1MF, are more positively associated with the NHEJ repair deficient class, with -1T having the highest odds ratio from all the features and -1C having the smallest odds ratio from all the encoding features. Inversely, G's appear more negatively associated, whereas A's appear to fluctuate depending on their position. Thirdly, the PAM sequence between +3 and +5 shows that the GG sequences do not influence the score since these are constant throughout all sequences. In conclusion, these odds ratio trends appear to be described in the 1MF encoding, except for the positional information, which does not appear to impact the accuracy, as shown in Figure 12.

Looking at the FPI for the RF model when using DS-Ins-HL & OHE, we see that most of the OHE features had little to no impact on the resulting score. The small deletion size features had the most impact, and the OHE features had an average decrease in score of approximately 0.00001, irrespective of their position, except for the GG sequence in the PAM sequence, with a decrease in score of zero.

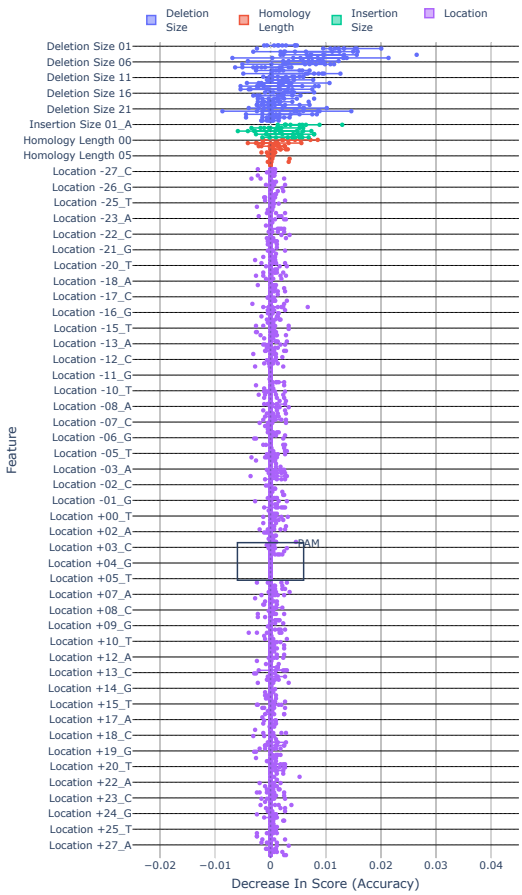


Fig. 33: The feature permutation importance on the validation set for DS-Ins-HL & OHE. The OHE features have little to no impact on the score.

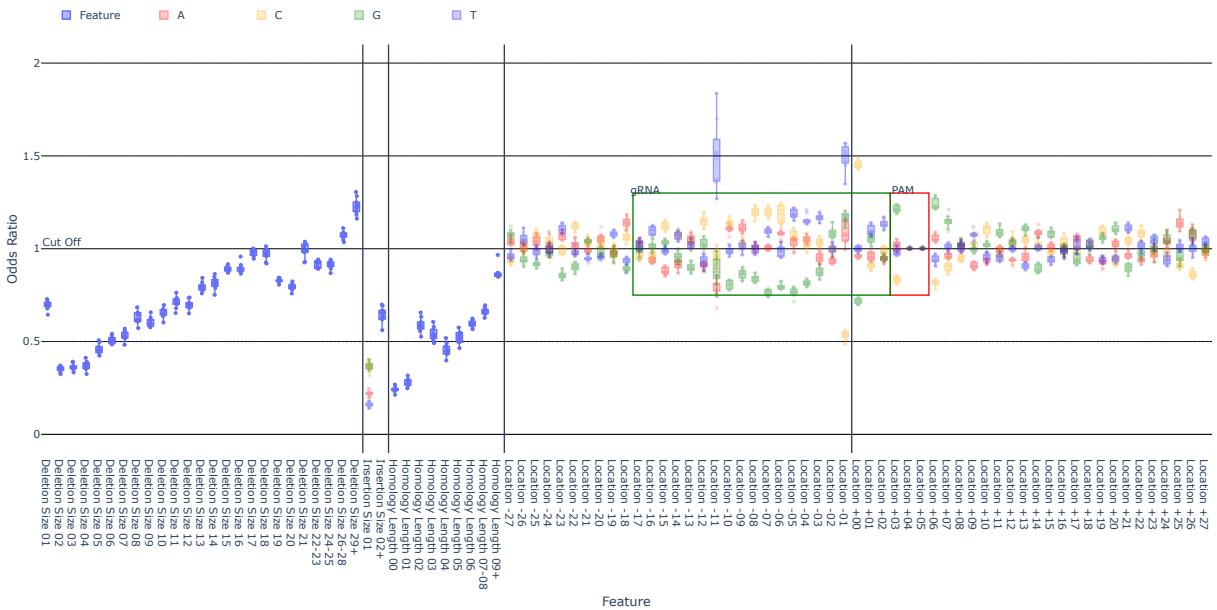


Fig. 32: The odds ratio values for DS-Ins-HL & OHE. The OHE features appear to have a negative/positive association with class based on their position.

E.3. One-Hot Encoding and 1-mer Frequency Encoding

Apart from the k-mer frequency encoding and OHE, we also evaluated the OHE alongside the 1-mer frequency. Figure 12 shows a significant improvement when adding OHE and 1-MF to DS-Ins-HL (LR: $p = 0.683$, RF: $p = 0.031$). Therefore, we look into the odds ratios and FPI to learn the trends these models capture.

Looking at the Odds Ratios in Figure 35, we observe the same observations we saw in Appendix E.2, where positional information contributes to the magnitude of the classes. In addition to those, we see the same behaviour we saw with the 1MF odds, where if a target sequence has a higher A or G content on the forward strand, it is more likely that the NHEJ repair pathway will be utilised. However, in this case, T's and G's are more discriminative of the class than A's and C's due to the difference in the odds ratio value. In conclusion, these odds ratio trends appear to describe the same trends observed under 1MF and OHE independently.

Looking at the FPI for the RF model when using DS-Ins-HL & OHE-1MF, we see that most of the OHE features had little to no impact on the resulting score, depicted in Figure 34. With an average decrease in score of approximately 0.00001, irrespective of their position, except for the GG sequence in the PAM sequence, with a decrease in score of zero. The 1MF features retained the same trends we saw originally and were still more impactful on the score than the OHE features.

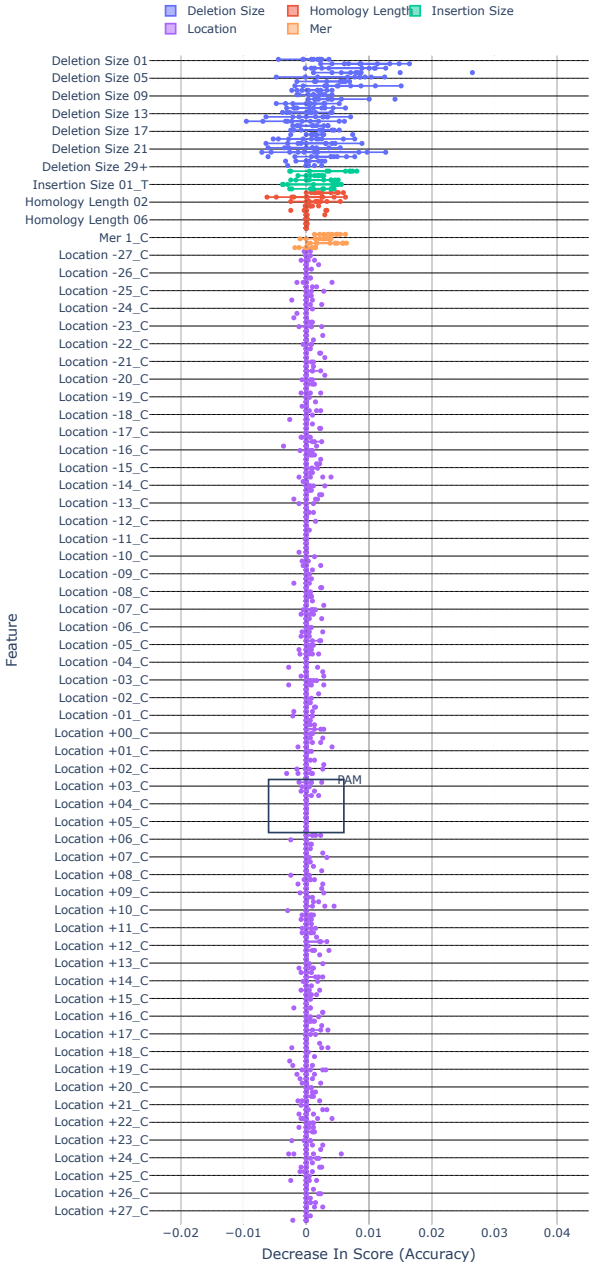


Fig. 34: The feature permutation importance on the validation set for DS-Ins-HL & OHE-1MF. The OHE and 1MF features have little to no impact on the score.

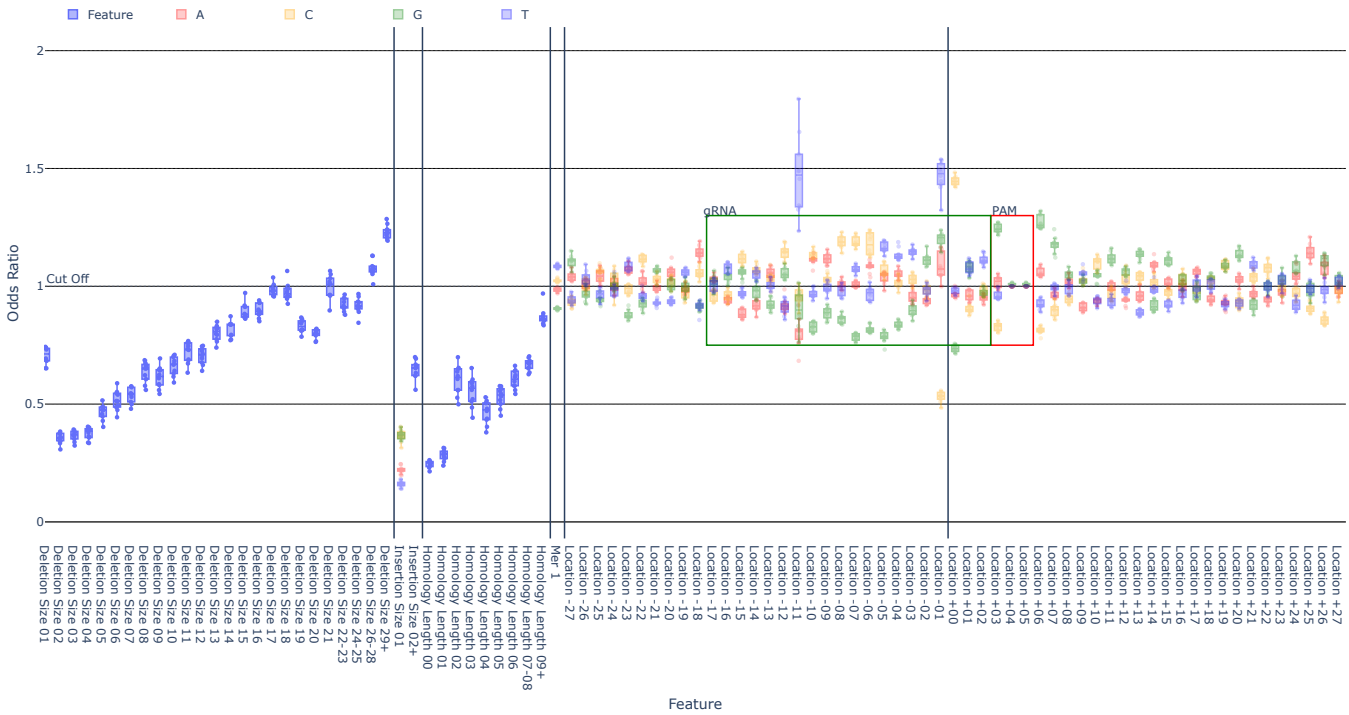


Fig. 35: The odds ratio values for DS-Ins-HL & OHE-1MF. The OHE features appear to have a negative/positive association with class based on their position.

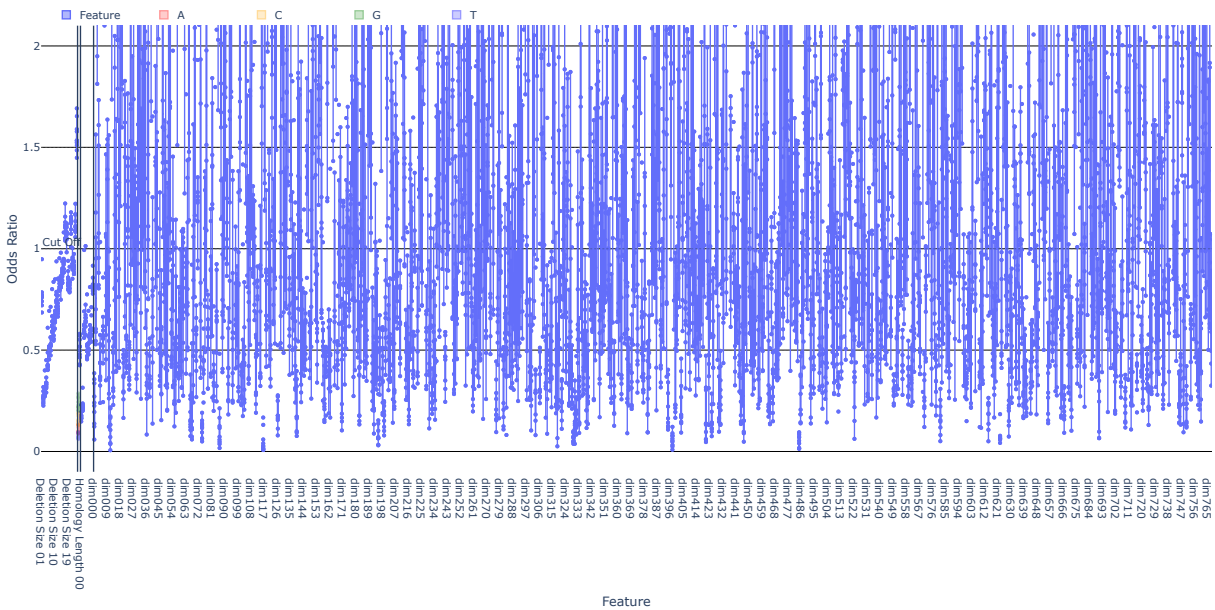


Fig. 36: The odds ratio values for DS-Ins-HL & DB3 - MP (`url`). The DNABERT mean pooling features resulted in noisy odd ratio values, fluctuating between $\approx 1 \times 10^{-20}$ and $\approx 1 \times 10^{20}$

E.4. DNABERT with Mean Pooling

The initial trends observed on frequency distribution features remained the same when evaluating the odds ratios for DB3 using the LR. As depicted in Figure 36, small deletion sizes, insertions and homology lengths remained discriminative towards the original classes reported. In contrast, newly introduced representation features resulted in noisy odds ratio values.

When reviewing the FPI of the mean pooling, the decrease for the original features remained relatively similar to the FPI values reported by DS-Ins-HL. However, as depicted in Figure 37 for the representation features, the decrease in score was close to zero in most cases.

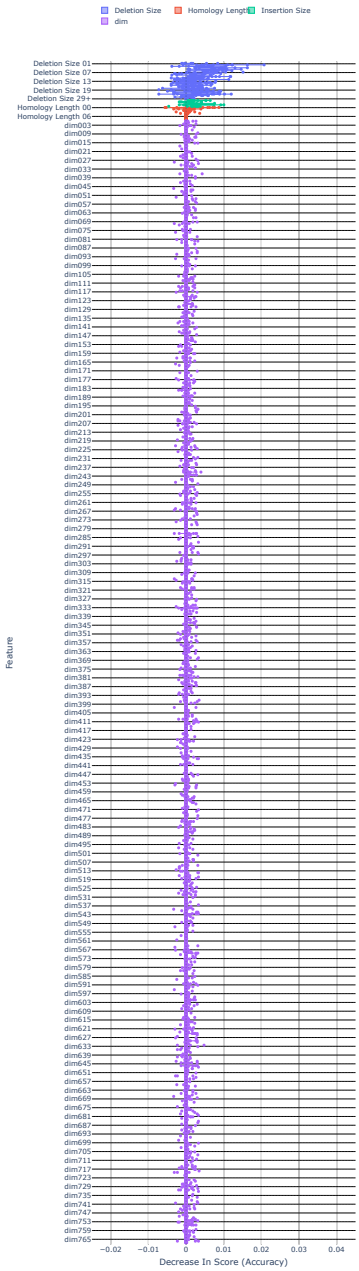


Fig. 37: The feature permutation importance on the validation set for DS-Ins-HL & DB3 - MP (`url`).

E.5. DNABERT with Horizontal Stacking

Similarly, with the horizontal stacking, the initial trends observed on frequency distribution features remained the same when evaluating the odds ratios for DB3 using the LR. As depicted in Figure 39, small deletion sizes, insertions and homology lengths are still the discriminative features of the class. In contrast, newly introduced representation features resulted in noisy odds ratios.

When reviewing the FPI of the horizontal stacking, the decrease for the original features remained relatively similar to the FPI values reported by DS-Ins-HL. However, as depicted in Figure 38 for the representation features, the decrease in score was close to zero in most cases.

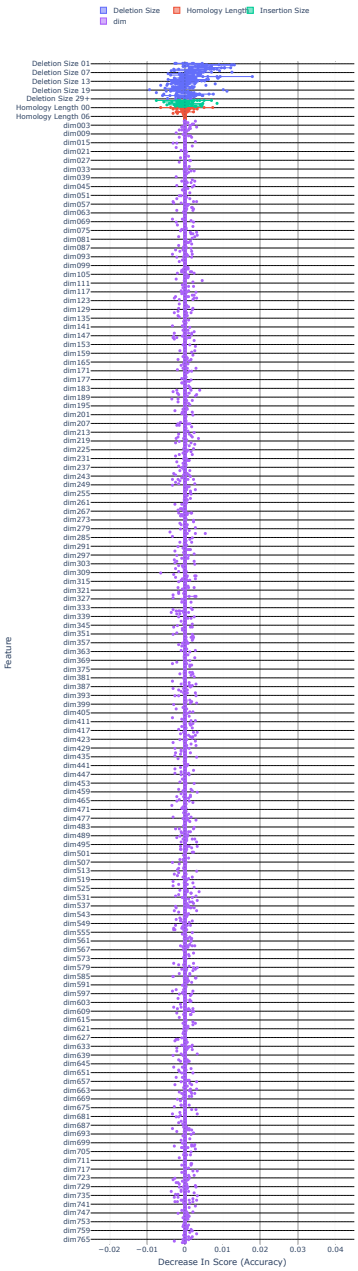


Fig. 38: The feature permutation importance on the validation set for DS-Ins-HL & DB3 - HS (`url`).

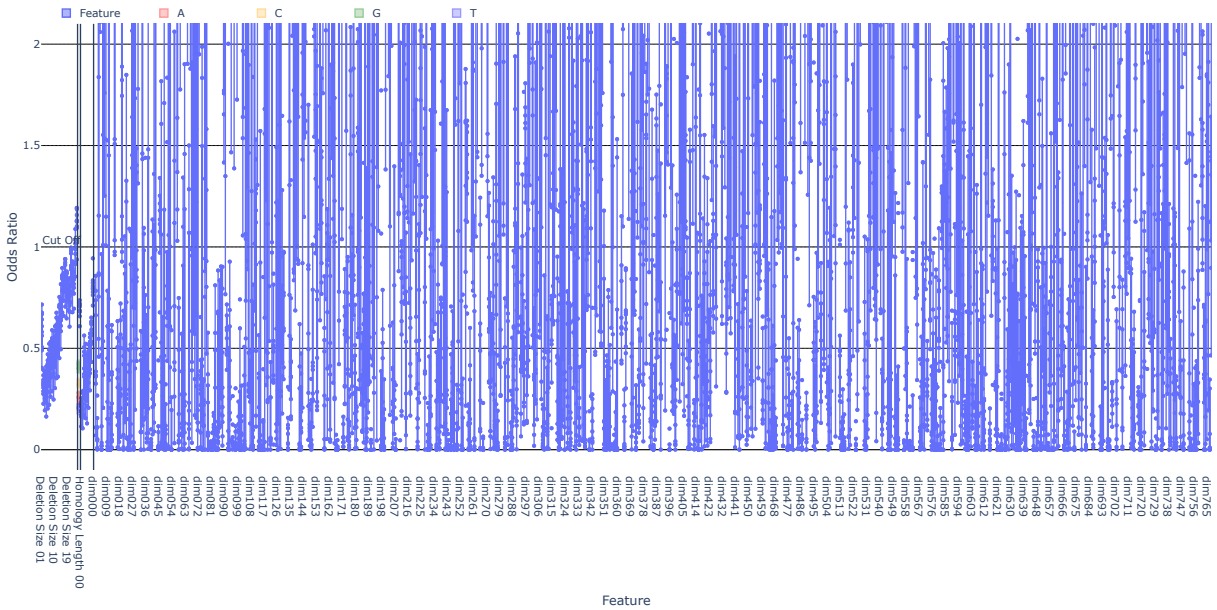


Fig. 39: The odds ratio values for DS-Ins-HL & DB3 - HS (`url`).

E.6. Supplementary Metrics

Even though accuracy is the metric we use to report the performance score, we also look at F1-Score, AUPRC, Precision, Recall, and AUROC. By evaluating these metrics, we ensure that the models are consistent with other performance metrics, allowing us to ensure robustness across different metrics.

Starting with Table 8, we show the performance scores for all six distribution groups. The distribution groups remained consistent across all performance scores, with DS-Ins-HL and SDS-Ins-HL reporting the best scores across all metrics. Focusing on the precision for these two, we see that DS-Ins-HL (LR: $78.46\% \pm 2.65\%$; RF: $92.36\% \pm 2.39\%$) observed a lower precision score ($p = 0.017$) than SDS-Ins-HL (LR: $80.09\% \pm 2.72\%$; RF: $91.70\% \pm 2.16\%$) in the LR model, yet the RF model shows that both distribution groups observe similar scores ($p = 0.241$).

Apart from the different metrics for the distribution groups, Table 9 describes the performance scores of all representations when used with DS-Ins-HL. For the encodings, we focus on 1-mer frequency encoding, one-hot encoding, and one-hot encoding with 1-mers. Whereas for the embeddings, we focus on vertical, horizontal and mean pooling, with a resulting dimensionality of 795 for VS or 768 for HS and MP. When reviewing the different metrics, they all follow the same trends we observe with accuracy. Mean pooling performs best from all representations, and horizontal stacking follows in second.

Table 8. The average accuracy, F1, the area under the precision-recall curve (AUPRC), precision, recall and receiver operating characteristic (AUROC) scores and standard deviation for our different distribution groups.

Model	Experiment	Accuracy	F1-Score	AUPRC	Precision	Recall	AUROC
Logistic Regression	MT[1]	$69.28\% \pm 2.38\%$	$70.20\% \pm 2.44\%$	$63.18\% \pm 2.04\%$	$68.15\% \pm 2.33\%$	$72.43\% \pm 3.33\%$	$75.95\% \pm 2.45\%$
	HL[2]	$68.78\% \pm 2.21\%$	$68.73\% \pm 2.43\%$	$62.97\% \pm 1.91\%$	$68.82\% \pm 2.15\%$	$68.68\% \pm 3.17\%$	$74.98\% \pm 2.18\%$
	DS-Ins[3]	$72.70\% \pm 2.34\%$	$71.93\% \pm 2.40\%$	$66.86\% \pm 2.38\%$	$74.05\% \pm 2.83\%$	$70.00\% \pm 2.95\%$	$78.47\% \pm 2.09\%$
	DS-Ins-HL[4]	$76.78\% \pm 2.02\%$	$76.08\% \pm 2.23\%$	$71.05\% \pm 2.20\%$	$78.46\% \pm 2.65\%$	$73.95\% \pm 3.54\%$	$83.15\% \pm 1.83\%$
	SDS-Ins[5]	$70.26\% \pm 2.72\%$	$68.51\% \pm 3.02\%$	$64.85\% \pm 2.72\%$	$72.89\% \pm 3.71\%$	$64.80\% \pm 4.26\%$	$75.64\% \pm 3.18\%$
	SDS-Ins-HL[6]	$77.57\% \pm 1.88\%$	$76.61\% \pm 1.88\%$	$72.12\% \pm 2.23\%$	$80.09\% \pm 2.72\%$	$73.49\% \pm 2.28\%$	$83.08\% \pm 1.59\%$
Random Forest	MT[1]	$76.64\% \pm 1.86\%$	$75.59\% \pm 2.20\%$	$71.11\% \pm 1.95\%$	$79.13\% \pm 2.07\%$	$72.43\% \pm 3.39\%$	$82.48\% \pm 1.95\%$
	HL[2]	$82.93\% \pm 1.78\%$	$82.36\% \pm 1.78\%$	$78.11\% \pm 2.26\%$	$85.27\% \pm 2.43\%$	$79.67\% \pm 1.90\%$	$89.56\% \pm 1.05\%$
	DS-Ins[3]	$93.85\% \pm 1.39\%$	$94.01\% \pm 1.31\%$	$90.26\% \pm 2.12\%$	$91.77\% \pm 2.00\%$	$96.38\% \pm 1.09\%$	$98.14\% \pm 0.86\%$
	DS-Ins-HL[4]	$94.44\% \pm 1.39\%$	$94.59\% \pm 1.29\%$	$91.07\% \pm 2.30\%$	$92.36\% \pm 2.39\%$	$96.97\% \pm 1.04\%$	$98.26\% \pm 0.81\%$
	SDS-Ins[5]	$93.26\% \pm 1.75\%$	$93.45\% \pm 1.65\%$	$89.42\% \pm 2.63\%$	$91.04\% \pm 2.70\%$	$96.05\% \pm 2.15\%$	$97.70\% \pm 0.86\%$
	SDS-Ins-HL[6]	$93.55\% \pm 1.09\%$	$93.70\% \pm 1.02\%$	$89.95\% \pm 1.84\%$	$91.70\% \pm 2.16\%$	$95.86\% \pm 1.70\%$	$98.03\% \pm 0.72\%$

[1]Mutation Type Frequency

[2]Homology Length Frequency Distribution

[3]Deletion Size and Insertion Frequency Distribution

[4]Deletion Size, Insertion and Homology Length Frequency Distribution

[5]MH Deletion Size, MH-Less Deletion Size and Insertion Frequency Distribution

[6]MH Deletion Size, MH-Less Deletion Size, Insertion and Homology Length Frequency Distribution

Table 9. The average accuracy, F1, the area under the precision-recall curve (AUPRC), precision, recall and receiver operating characteristic (AUROC) scores and standard deviation for the feature group (DS-Ins-HL) and when this feature group is used with different representations.

Model	Experiment	Accuracy	F1-Score	AUPRC	Precision	Recall	AUROC
Logistic Regression	DS-Ins-HL	76.78% ± 2.02%	76.08% ± 2.23%	71.05% ± 2.20%	78.46% ± 2.65%	73.95% ± 3.54%	83.15% ± 1.83%
	DS-Ins-HL & 1MF[1]	77.07% ± 1.60%	76.38% ± 1.99%	71.35% ± 1.68%	78.78% ± 2.36%	74.28% ± 4.04%	83.28% ± 1.83%
	DS-Ins-HL & OHE[2]	77.04% ± 1.61%	76.23% ± 1.71%	71.40% ± 1.81%	79.04% ± 2.27%	73.68% ± 2.56%	82.88% ± 1.75%
	DS-Ins-HL & OHE-1MF[3]	77.04% ± 1.61%	76.23% ± 1.71%	71.40% ± 1.81%	79.04% ± 2.27%	73.68% ± 2.56%	82.88% ± 1.75%
	DS-Ins-HL & DB3[4] VS[5]	69.34% ± 1.81%	69.14% ± 1.81%	63.52% ± 1.70%	69.69% ± 2.59%	68.75% ± 3.39%	75.96% ± 1.42%
	DS-Ins-HL & DB3[4] HS[6]	77.04% ± 1.88%	76.27% ± 2.27%	71.34% ± 1.86%	78.84% ± 1.75%	73.95% ± 3.63%	83.05% ± 1.88%
	DS-Ins-HL & DB3[4] MP[7]	68.29% ± 2.26%	68.30% ± 2.49%	62.55% ± 2.09%	68.36% ± 2.92%	68.42% ± 4.21%	74.56% ± 1.93%
Random Forest	DS-Ins-HL	94.44% ± 1.39%	94.59% ± 1.29%	91.07% ± 2.30%	92.36% ± 2.39%	96.97% ± 1.04%	98.26% ± 0.81%
	DS-Ins-HL & 1MF[1]	94.80% ± 1.43%	94.93% ± 1.36%	91.66% ± 2.27%	92.93% ± 2.21%	97.04% ± 1.25%	98.57% ± 0.65%
	DS-Ins-HL & OHE[2]	94.67% ± 1.52%	94.83% ± 1.43%	91.28% ± 2.51%	92.35% ± 2.56%	97.50% ± 1.31%	98.40% ± 0.79%
	DS-Ins-HL & OHE-1MF[3]	95.07% ± 1.04%	95.20% ± 0.98%	91.83% ± 1.74%	92.79% ± 1.75%	97.76% ± 0.71%	98.61% ± 0.68%
	DS-Ins-HL & DB3[4] VS[5]	94.28% ± 0.98%	94.45% ± 0.94%	90.64% ± 1.51%	91.77% ± 1.40%	97.30% ± 0.85%	98.49% ± 0.71%
	DS-Ins-HL & DB3[4] HS[6]	95.39% ± 1.48%	95.50% ± 1.41%	92.52% ± 2.40%	93.61% ± 2.32%	97.50% ± 1.11%	98.71% ± 0.55%
	DS-Ins-HL & DB3[4] MP[7]	96.12% ± 1.67%	96.22% ± 1.60%	93.47% ± 2.67%	94.19% ± 2.46%	98.36% ± 1.09%	98.89% ± 0.54%

[1]1MF represents using the 1-Mer frequency distribution. The 1MF adds 4 dimensions (one for each nucleotide) to the original experiment.

[2]OHE represents using the one-hot encoding features. The OHE adds 220 dimensions to the original experiment.

[3]OHE-1MF represents using the one-hot encoding features and the 1-Mer frequency features. The OHE adds 220 dimensions to the original experiment, while the 1MF adds another 4 features.

[4]DB3 represents using the pre-trained DNABERT, with a k-mer of 3. The DB3 concatenates 768 dimensions to the original experiment.

[5]VS represents using the vertical stacking technique. This technique has a resulting dimensionality of 795 (15 dimensions for each token).

[6]HS represents the horizontal stacking technique. This technique uses UMAP and results in a dimensionality of 768.

[7]MP represents using the mean pooling technique. This technique has a resulting dimensionality of 768 dimensions.

E.7. Test Set Score

As discussed in Section 2.5, we created a held-out test set. This test set consists of previously unseen target sequences and mutations. This allows us to see how the model performs on unseen samples and ensures that the conclusions derived from this study can be made to new samples.

As shown in all three figures (Figure 40, Figure 41, and Figure 42), the LR model reports no variance; since this model converges to a solution, re-running multiple times will not yield different results. In contrast, we can see some variance in the results reported in the RF model, which uses bagging and randomly selects features.

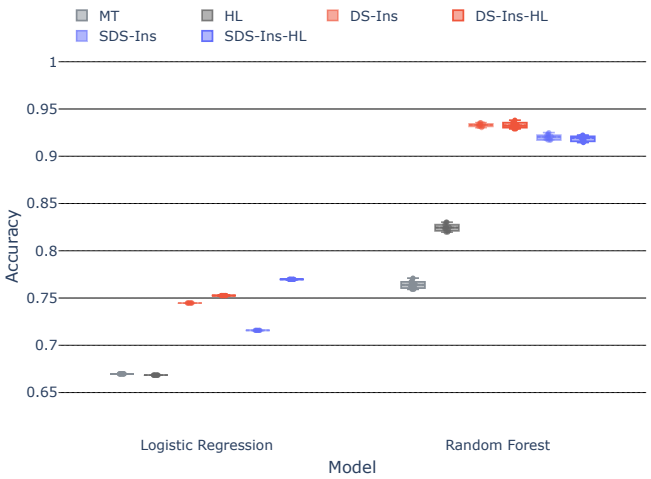


Fig. 40: Test accuracy scores for all different feature groups. Each point represents the test score achieved when running the model multiple times (same train/test datasets). The same trends observed with the train and validation sets persist. SDS-Ins-HL is the best-performing distribution group for LR and DS-Ins-HL performs the best when using RF.

Starting with Figure 40, we can see the performance of the different distribution groups. From this, we can see that the same trends we observed during cross-validation are persistent on the test set. DS-Ins-HL still provided the best performance from all groups. This can be observed because deletion size is the most important distribution, and splitting it by homology type does not improve the performance. Similarly, using homology length improved the performance of the LR model.

Focusing on the encoding representations with DS-Ins-HL, as shown in Figure 41, we can see that 1MF and OHE-1MF encoding still performed the best with the RF model. When comparing the two encoding techniques, we can see little to no difference between them, thus further confirming the suggestion for using 1MF as the preferred encoding technique.

Figure 42 shows the embedding representations used with DS-Ins-HL. The same trends we saw with the cross-validation resurface with the test set. Focusing on the RF model, we see that HS and MP still outperform all of the other representations, with the only slight difference being that HS provided a slightly higher accuracy rate than that we saw with the MP technique. The LR model saw a drop in performance for the VS and MP techniques, and HS provided no improvement.

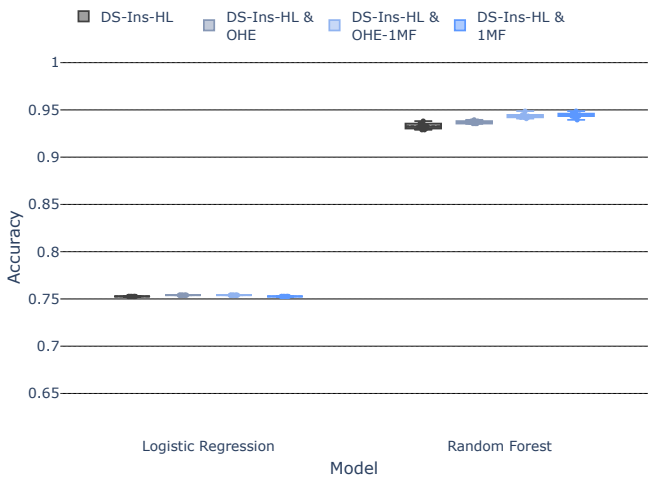


Fig. 41: Test accuracy scores for the DS-Ins-HL feature group and all the encoding techniques. The same trends observed with the train and validation sets persist. 1MF frequency encoding yields the highest accuracy for the RF model, with no difference between performs 1MF and OHE-1MF.

Finally, the accuracy scores in all the Figures depicted are slightly lower than the cross-validation scores. However, comparing both data sets, they observe similar mean and median values.

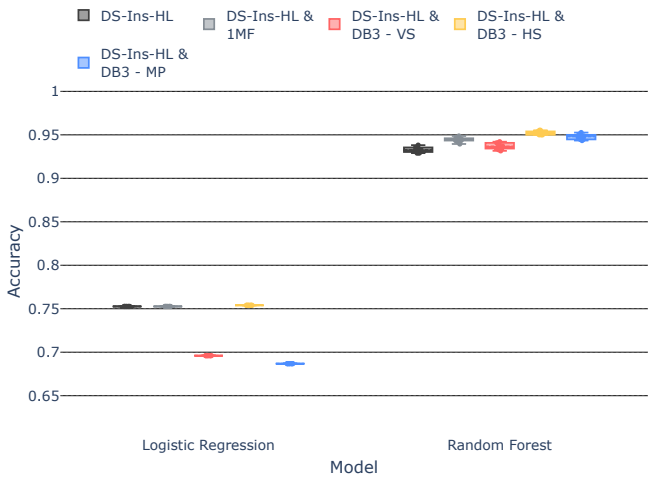


Fig. 42: Test accuracy scores of DS-Ins-HL, DS-Ins-HL & 1MF, DS-Ins-HL with vertical (VS), horizontal stacking (HS), and mean pooling (MP). HS and MP features have a dimensionality of 768. VS has a dimensionality of 795 (15 dimensions for each token). Both HS and VS are generated using UMAP as the dimensionality reduction technique. The same trends observed with the train and validation sets persist. LR observes a drop was seen when using VS and MP. RF observes an improvement when using HS and MP, but no changes in the accuracy when using VS.

F. DNABERT

When working with the embedding techniques, the DNABERT model and the stacking functionality consist of several parameters which impact the resulting performance scores. In this appendix, we highlight a number of things, and we start by comparing pre-training and fine-tuning and the generation of the DNABERT tokens. After which, we detail the different dimensionality reduction techniques, the variable dimensional space of VS and HS and the different k-mer values.

When comparing pre-training and fine-tuning, the differences are subtle. The pre-training model learns a general understanding of the DNA sequences presented through unlabelled data. The fine-tuned model uses this learned general understanding, optimises it using application-specific labelled data, and predicts a problem.

During the pre-training phase, the model learns the weights of the attention layers based on the large corpus of sequences. DNABERT tokenises the entire human genome, comprising approximately 5 million target sequences varying in length (between 5 and 512 nucleotides). Following the tokenisation process, a percentage of the k-mers in each sequence are masked. The model then attempts to reconstruct the original k-mer using the neighbours of the masked tokens.

During the fine-tuning process, the previously pre-trained model initialises the model weights. Afterwards, application-specific labelled data is tokenised but not masked. Using these tokens, it tries to predict the label [7]. The fine-tuning functionality allows other researchers to use the pre-trained model and adapt it to their specific problem when sequences are labelled.

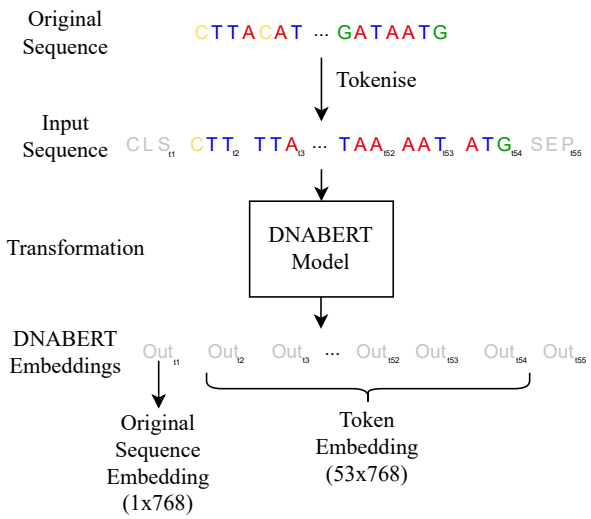


Fig. 43: The process of generating DNABERT Embeddings. The original target sequence is passed through a parser, creating tokens (k-mers) and appending the classification and separation tokens. The input sequence is then passed to a DNABERT pre-trained model, where the positional and token embeddings are generated and updated based on the model's weights. Finally, the last hidden state of the attention layer is used to extract the DNABERT embeddings, where the first token is associated with the CLS token, the last token is associated with the SEP token, and everything else is the k-mer embeddings.

To generate and extract the DNABERT token embeddings, we use the process depicted in Figure 43. Initially, we start with a target sequence made up of 55 nucleotides. This sequence is then split into overlapping k-mers (total of n), and then CLS tokens and SEP tokens are appended to the start and end of the k-mers, respectively. These tokens are then fed into a pre-trained DNABERT model, which generates the embeddings using the learned weights. These outputted embeddings return one embedding describing the CLS token, a sequence representation and all the token embeddings, where each embedding describes the respective k-mer. Following this embedding generation process, we use all the token embeddings (53). These are utilised by applying a reduction technique like vertical, horizontal or mean pooling to obtain a smaller feature space and avoid the curse of dimensionality.

F.1. PCA as dimensionality reduction

We reduce the feature space through the dimensionality reduction technique when working with the vertical and horizontal stacking techniques. We report the UMAP results as this technique can capture complex non-linear relationships found in the feature space, which PCA cannot.

In Figure 44, we see the performance of horizontal stacking when using PCA and UMAP contrasted against each other. Across the multiple dimensions we evaluated, we see that UMAP provided stronger accuracy values. As the resulting dimensionality increased, PCA saw a drop in the performance for the LR model and an increase in variance (with no improvement) for the RF model. In contrast, UMAP maintained its performance across both models, with LR observing a slight increase in the variance reported. The RF observed significant improvements across all dimensions when comparing both reduction techniques.

Figure 45 shows the performance of vertical stacking when using PCA and UMAP, contrasted against each other. Across the multiple dimensions we evaluated, both dimensionality reduction

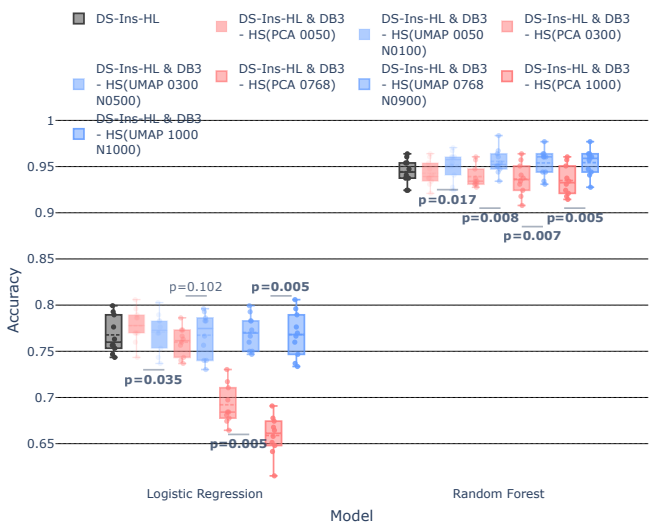


Fig. 44: Validation accuracy score of DS-Ins-HL when using Horizontal Stacking (`url`). Those shaded in red are calculated using PCA as a dimensionality reduction, whereas those in blue are calculated using UMAP as a dimensionality reduction technique.

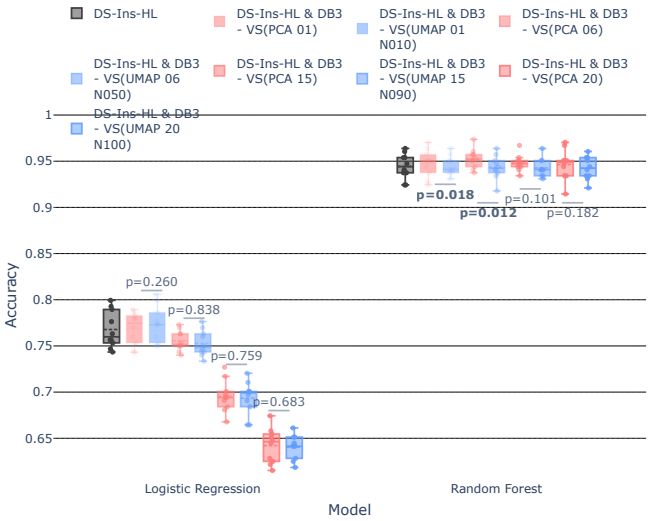


Fig. 45: Validation accuracy score of DS-Ins-HL when using Vertical Stacking (`url`). Those shaded in red are calculated using PCA as a dimensionality reduction, whereas those in blue are calculated using UMAP as a dimensionality reduction technique.

techniques performed similarly. As the resulting dimensionality increased, PCA and UMAP saw a drop in the performance for the LR model. In contrast, the RF maintained its performance across both techniques, with none of the dimensions reporting a significant improvement from the baseline.

PCA allows us to examine the explained variance in the components. Table 10 shows the variance explained when using vertical stacking at the different dimensions. This describes how much of the variance of a single token we are describing. Whereas Table 11 shows the variance explained when using horizontal stacking. This described how much of the variance of all the tokens we are describing.

Table 10. Cumulative explained variance for the vertical stacking.

Number of Components	3-mer	4-mer	5-mer	6-mer
1	5.68%	4.04%	3.28%	3.76%
2	10.45%	7.36%	6.29%	7.24%
6	27.10%	18.92%	16.87%	17.90%
10	39.80%	27.33%	23.46%	23.96%
15	51.58%	35.12%	30.16%	30.38%
16	53.58%	36.45%	31.40%	31.52%
20	60.27%	41.01%	36.15%	36.01%

As shown, the dimensionality reduction for the vertical stacking struggles to describe the variance in all the token embeddings. Focusing on the overall cumulative explained variance of these k-mers, even when using 20 components (resulting in 1,060 dimensions), the model struggles across all the different k-mers, achieving a maximum of 60.27% explained variance. Looking at the different k-mers, we can see that as the k increases, the PCA struggles to explain the variance in the entire feature space, with 6-mers observing the worst explained variance across all

Table 11. Cumulative explained variance for the horizontal stacking.

Number of Components	3-mer	4-mer	5-mer	6-mer
50	30.23%	24.18%	21.00%	21.49%
100	44.78%	36.00%	32.50%	33.88%
300	72.82%	60.60%	59.84%	61.86%
500	84.84%	73.87%	75.07%	77.13%
750	91.46%	84.37%	86.08%	88.00%
768	91.76%	84.95%	86.67%	88.57%
1000	94.81%	91.01%	92.52%	93.96%

k-mers. This means that PCA struggles to describe the non-linear relationships enclosed within the token embeddings.

When focusing on the variance explained when using the horizontal stacking, we can see that PCA could describe 94.81% cumulative explained variance with 1000 components (for 3-mer) over all the token embeddings. The cumulative variance is relatively low when looking at the different components. Even though the model could achieve high cumulative explained variance, it struggled with smaller components. This means the variance is not focused on a single component but rather throughout the reduced space. Focusing on the different k-mers, we see a similar trend as the vertical stacking. As the k-mer size increased, PCA struggled to describe the relationships within the stacked token embedding space. This reinforces our statement that UMAP provides a more suitable dimensionality reduction technique for this problem.

F.2. Horizontal Stacking with different dimensionality

Apart from evaluating the different dimensionality techniques, we researched the impact of different resulting dimensionalities. Unlike mean pooling, where the dimensionality always results in 768, for horizontal stacking, the resulting dimensionality is a parameter that can change to reflect the number of samples available or the application at hand.

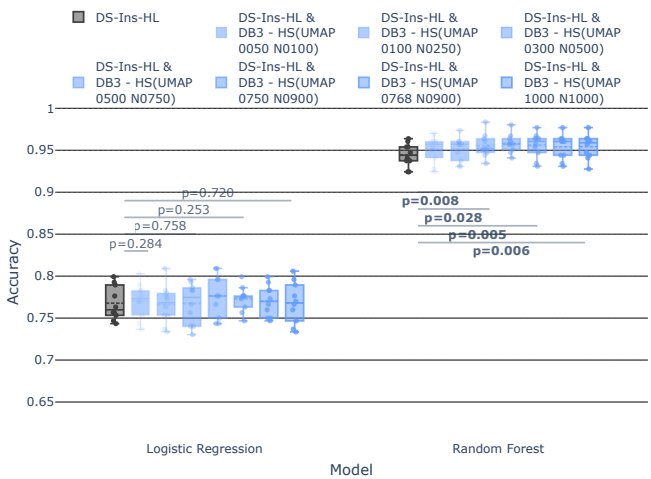


Fig. 46: Validation accuracy score of DS-Ins-HL when using different Horizontal Stacking dimensions (`url`). The dimensions are reduced using UMAP.

Figure 46 shows the different resulting dimensionalities, starting with only 50 features to describe the token embeddings and ending with 1000 features. As shown, the LR reports no significant differences with any dimensionality. In contrast, the RF model sees a statistical difference even when using 50 dimensions to describe the token embeddings (originally 40,704). From these, we can see that 500 dimensions yielded the lowest variance; however, the mean and median values reported were similar for all resulting dimensionality.

F.3. Vertical Stacking with different dimensionality

Similar to horizontal stacking, vertical stacking can reduce the dimensionality at the token level. This means that the dimensionality of each token in the representations is reduced to a specific dimensionality. Afterwards, the resulting dimensionality is equivalent to the number of tokens multiplied by the dimensionality used.

Figure 47 shows the different token dimensionalities used, where to obtain the resulting dimensionality, we multiply by 53 (total number of tokens). The lowest dimension is one dimension per token (resulting in 53 features), and the largest is 20 dimensions per token (resulting in 1060 features). As shown, the LR reports a significant decrease in the accuracy as the number of dimensions to describe each token increases. In contrast, the RF model sees no statistical difference with any dimensionality, with all the different runs reporting a p-value higher than 0.05. From these, we can see that vertical stacking features did not aid the prediction model, meaning that during the stacking process, vital information that is retained during horizontal stacking is lost with vertical stacking.

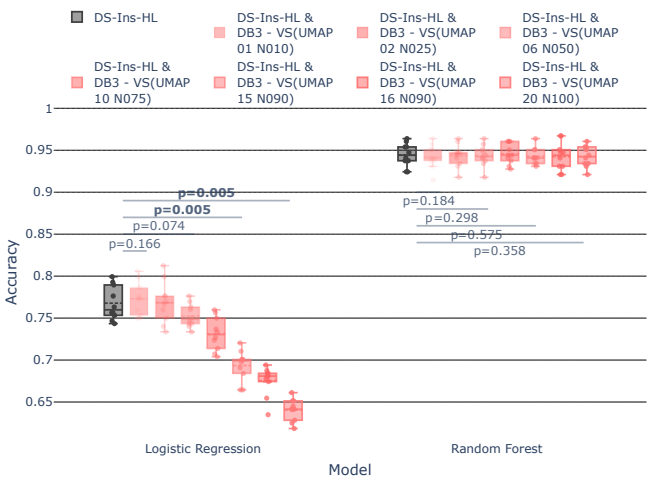


Fig. 47: Validation accuracy score of DS-Ins-HL when using different Vertical Stacking dimensions (`url`). The dimensions are reduced using UMAP.

F.4. DNABERT using k-mers of size 4

Ji et al. [7] published four pre-trained models for different k-mer values, specifically 3, 4, 5, and 6. Our research focused on 3-mers since there were no significant improvements when using any other k-value. In addition, 3-mers allow for shorter sequences, possibly easier to understand and interpret.

In Figure 48, 3-mer and 4-mers are compared using different stacking techniques. No significant difference is observable when using any stacking/pooling techniques. There is a non-significant change in the inner quartile of VS for both models, but p-values are higher than the 0.05 confidence level (LR: $p = 0.139$, RF: $p = 0.305$).

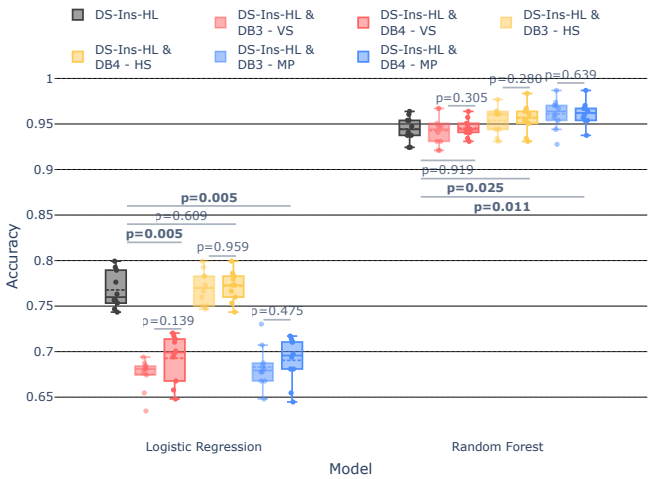


Fig. 48: Validation accuracy score of DS-Ins-HL, DS-Ins-HL with vertical (VS), horizontal stacking (HS), and mean pooling (MP) when using DNABERT with 3-mer and 4-mer (`url`). HS and MP features have a dimensionality of 768. VS has a dimensionality of 15 dimensions for each token (DB3=795; DB4=780). Both HS and VS are generated using UMAP as the dimensionality reduction technique. LR and RF models do not report a significant difference among any stacking techniques.

F.5. DNABERT using k-mers of size 5

The 5-mers followed a similar trend to the 4-mers, as shown in Figure 49. When comparing the 3-mers and 5-mers against each other, no technique reports a significant difference in the results. In addition, the different stacking techniques still yielded similar observations for both models when compared against the original DS-Ins-HL distribution group. In the RF model, VS did not see any differences from the original distribution group. HS and MP saw a slight improvement, with MP performing slightly better amongst both techniques (HS : $95.86\% \pm 1.51\%, p = 0.009$; MP : $95.99\% \pm 1.36\%, p = 0.009$). In contrast, for the LR model, VS and MP saw a drop in performance (VS : $68.29\% \pm 1.80\%, p = 0.005$; MP : $69.18\% \pm 1.68\%, p = 0.005$), and HS saw no significant differences ($77.14\% \pm 2.39\%, p = 0.200$).

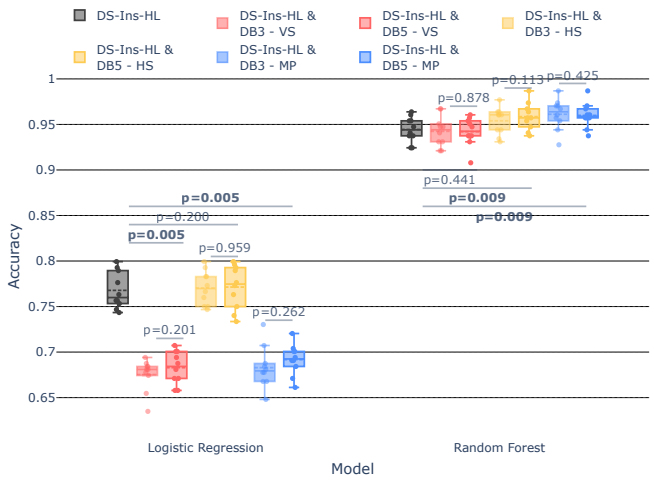


Fig. 49: Validation accuracy score of DS-Ins-HL, DS-Ins-HL with vertical (VS), horizontal stacking (HS), and mean pooling (MP) when using DNABERT with 3-mer and 5-mer (`url`). HS and MP features have a dimensionality of 768. VS has a dimensionality of 15 dimensions for each token (DB3=795; DB5=765). Both HS and VS are generated using UMAP as the dimensionality reduction technique. LR and RF models do not report a significant difference among the stacking techniques.

F.6. DNABERT using k-mers of size 6

Finally, as shown in Figure 50, the 6-mers followed a similar trend to the other k-mers, with a slight difference for the VS and HS. Comparing the 3-mers and 6-mers against each other, HS and MP reported no significant difference in the results. However, VS saw a significant performance change for the RF model ($p = 0.041$), with the mean ($94.24\% \rightarrow 93.78\%$) and median ($94.41\% \rightarrow 93.75\%$) decreasing slightly and the standard deviation increasing ($0.0167 \rightarrow 0.0179$). Unlike what we previously reported when using the RF model, HS on DNABERT with 6-mers observed no significant improvement from the DS-Ins-HL ($94.44\% \pm 1.39\% \rightarrow 94.97\% \pm 1.66\%, p = 0.139$). MP saw a significant improvement, similar to what was observed with the 3-mers, $94.44\% \pm 1.39\% \rightarrow 95.62\% \pm 1.12\%, p = 0.009$.

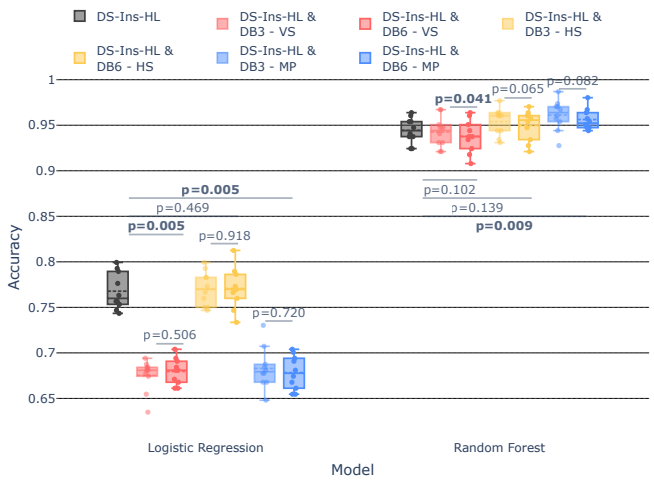


Fig. 50: Validation accuracy score of DS-Ins-HL, DS-Ins-HL with vertical (VS), horizontal stacking (HS), and mean pooling (MP) when using DNABERT with 3-mer and 6-mer (`url`). HS and MP features have a dimensionality of 768. VS has a dimensionality of 15 dimensions for each token (DB3=795; DB6=750). Both HS and VS are generated using UMAP as the dimensionality reduction technique. With LR, no significant differences are observable using

Supplementary References

1. Max W. Shen, Mandana Arbab, Jonathan Y. Hsu, Daniel Worstell, Sannie J. Culbertson, Olga Krabbe, Christopher A. Cassa, David R. Liu, David K. Gifford, and Richard I. Sherwood. Predictable and precise template-free crispr editing of pathogenic variants. *Nature*, 563(7733):646–651, 2018.
2. Benjamin M. Stinson and Joseph J. Loparo. Repair of dna double-strand breaks by the nonhomologous end joining pathway. *Annual review of biochemistry*, 90(1):137–164, 2021.
3. Brandi L. Mahaney, Michal Hammel, Katheryn Meek, John A. Tainer, and Susan P. Lees-Miller. Xrcc4 and xlf form long helical protein filaments suitable for dna end protection and alignment to facilitate dna double strand break repair. *Biochemistry and cell biology*, 91(1):31–41, 2013.
4. Ruben Dewitte. The do's and dont's of data binning for distributional analysis. `https://rubendewitte.be/blogposts/DataBinning.html`, 2020. Accessed: 2023-08-31.
5. Herbert A. Sturges. The choice of a class interval. *Journal of the American Statistical Association*, 21(153):65–66, 1926.
6. David P Doane. Aesthetic frequency classifications. *The American Statistician*, 30(4):181–183, 1976.
7. Yanrong Ji, Zhihan Zhou, Han Liu, and Ramana V Davuluri. DNABERT: pre-trained Bidirectional Encoder Representations from Transformers model for DNA-language in genome. *Bioinformatics*, 37(15):2112–2120, 02 2021.