Detecting & Extracting Information from Genetically-Mutated Data Using AI

Article

Keywords:
Neural Networks, Frameshift Mutation, Substitution Mutation

Akhil Ganesan*

*Georgia Institute of Technology
Email: akhilganesan526@gmail.com

Abstract

The capacity for AI to revolutionize data analysis is limitless. This study explores generalized datasets which have defects analogous to genetic mutations (specifically, frameshift & substitution mutations) by attempting to use neural networks to predict whether frameshift mutations have occurred (which it could with around 95% accuracy) and prior to the frameshift, the probability substitution mutations occur (which it could with around 80% accuracy). While this is most intuitively applicable to genetic data, the applications are wide-ranging beyond this as numerous other types of data defects can be described using substitution & frameshift mutation analogs.

Introduction

Artificial Intelligence (AI) has time & time again proven to surpass traditional methods of data analysis. In this study, we apply AI towards a biocomputing avenue: analyzing frameshift mutations. In genetics, a frameshift mutation consists of either an insertion or deletion of a nucleotide, resulting in a shift of the codon reading frame for preceeding nucleotides, drastically changing the translated protein typically causing malfunction. This varies from a substitution mutation, which changes a single nucleotide's value, leaving the preceeding nucleotide positions unchanged. However, the concept of a frameshift and substitution mutation aren't just present in genetic datasets; for instance, substitutions and frameshifts can occur in student test-taking data when answers are inputed (i.e. bubbling incorrect answers are substitution, bubbling answers in the wrong question number/location are frameshift). This paper details an attempt to develop a neural network architecture trained for 2 purposes:

1. Detecting generalized frameshift defects in data (with substitution and frameshift mutations variably present)
2. Deriving information of the original source from the defective data

While the first objective is expected to be easier to achieve due to a frameshift's cascading effects on the dataset, the second objective's success may be more sensitive to dataset factors and the specific information being derived. These variables are defined for our study, but may be a point of manipulation for future studies.

Methods

This study involved 2 main phases: generating the dataset & developing the AI. Before discussing the procedure, the information we will attempt to derive for objective 2 is the probability of substitution mutations occurring prior to a frameshift.

Generating the Dataset

This study uses randomly generated data rather than genomic datasets to better emulate a frameshift mutation for generic data. The following variables are present for manipulation:

- C_l: The key length; the experimental values varied between 10 and 100
- C_n: The number of data values possible (analagous to 4 nucleotides possible in DNA or RNA); the experimental value was 4
- C_m: The probability that a substitution mutation occurs multiple times; the experimental value was 80% (first objective) & varied between $100\%/C_n$ and 95% (second objective)
- C_f: The probability that a frameshift mutation occurs multiple times (note the frameshifted data always has at least 1 frameshift); the experimental value was 80%

The data was generated in the following sequence:

1. **Develop a Key**: Randomly generate a sequence of numbers (ranging from 1 to C_n) with a length C_l
2. **Add Substitution Mutations**: Using the probability C_m, modify numbers in the key to generate an answer list
3. **Add Frameshift Mutations**: Using a probability C_f, modify the answer list by adding insertions and/or deletions (where insertions remove the last element and deletions add another random element to the end to preserve length)
4. **Generate a Checked List**: Compare the defected answer list to the original key, marking 1s where numbers are the same and 0 where the numbers vary
5. **Create Test & Training Datasets for Objective 1**: Iteratively generate a key and add 2 checked lists (one with only substitution mutations, the other with both) to the input dataset, adding 0 & 1 to the output dataset (representing whether a frameshift occurred or not). Then partition the first 80% of both the input & output set for training, leaving the remaining 20% for testing.
6. **Create Test & Training Datasets for Objective 2**: Repeat the above process for varying C_m values to generate the input & output sets (input contains checked lists, output contains C_m), again partitioning a 80% of the data (randomized by C_m value) for training & 20% for testing

Developing the AI

A neural network architecture was chosen as objective 1 was a classification problem & objective 2 was a regression problem within limits (accuracy between 0 & 1). While this didn't impact the structure of the network, it caused differences in their training evaluators: The first network trained using the Adam optimizer with binary crossentropy loss while the second network trained using stochastic gradient descent based on a mean absolute error cost function. The code for the network is present on the project github[1].

Results

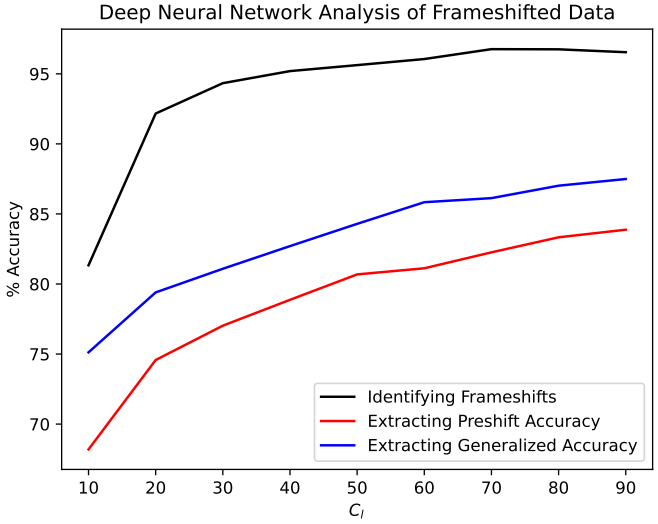

Figure 1. Graph of C_l v.s. % Accuracy

Discussion

The major result observed was as C_l increased, the shift prediction & C_m prediction accuracy increased. This result also makes intuitive sense as the cascading effects of a frameshift mutation are more likely to be differentiated from single-instance substitution mutations across larger-sized data. However, every accuracy across the test cases were all fairly high (above 70% for most) and may be sufficient depending on the specific application.

When attempting to integrate these 2 tasks to develop a single AI that predicts C_m from both only substituted & substituted/frameshifted data (labeled as Extracting Generalized Accuracy on the graph), the overall accuracy achieved was between that of both tasks individually.

Conclusion

As expected, the network performed better on detecting frameshift mutations compared to deriving information about the original data. Regardless, both tasks' accuracy values show neural networks have great potential in achieving both objectives if implemented with greater complexity specialized for the task. However, some limitations observed (and possible areas for future exploration) were that detection of frameshifts was sharply limited as the data decreased in key-length, the accuracy could vary based on the number of possible nucleotides C_n, & the network only used checked lists rather than the raw nucleotide key/answer lists. The first limitation is intuitively expected as frameshifts are harder to differentiate from substitutions with fewer nucleotides; however, the rate at which the accuracy decreases with length could be further explored. The second & third limitation go hand in hand: in this study, the network only trained on the checked lists. If the C_n variable was modified, would it fundamentally change the structure of checked lists, or would the network have to input the raw nucleotide data for improved detection (if improved detection is a result)? This is likely the pathway to take to develop a network that isn't generalized, but rather has a database of common nucleotide sequences to compare input data with for gaining better insights.

Notes

1 github.com/akhil-ganesan/Frameshift-Mutation-Detector