VERTICALLY INTEGRATED

PROJECT (VIP) REPORT

NEUROMORPHIC SOLUTIONS TO DNA SUBSEQUENCE

AND RESTRICTION SITE ANALYSIS

*Ethan Jacob Moyer*

ENGR 370: Applications, Algorithms, & Architecture for

Neuromorphic Computing

*Dr. Anup Kumar Das*

Drexel University

27 February 2020

**INTRODUCTION**

BACKROUND:

Originally postulated by Watson and Crick in the 1950s, the process of cellular deoxyribonucleic acid (DNA) synthesis has been long understood as a trademark to biological studies (Cooper). Over the past few decades, understanding this process has led to the discovery and development of many synthetic manufacturing methods, namely syntenic gene synthesis (Hughes). As a result, other areas of genomics have accumulated massive research interest. From the start of the Human Genome Project in 1990 to the discovery of Cluster Regularly Interspaced Palindromic Repeat (CRISPR) Cas9 system in 2005, the field of genomics has proven to advance at incredible rates as the reservoir of information pertaining to DNA and gene research increases ("The Human Genome Project”) ("CRISPR Timeline"). Although the public adoption of genomic technologies and tools is arguably in the near feature, the cost of producing and researching novel genes remains as the field’s largest determinant (Tang). In a 2017 publication, two University of Texas laboratory and biological researchers, Randall A. Hughes and Andrew D. Ellington, assert that by decreasing the “DNA read-write cost gap,” humanity could claim this century as the “century of DNA.” Closing this gap will in fact allow researchers to more discover novel genes and genetic systems with far less resources. With the advent of such research, both industrial and academic disciplines involved may witness the advancement of certain fields including “novel therapeutics, materials, [and] biosensing” (Hughes).

One process that the two researchers highlight in their paper is oligonucleotide synthesis (also known as gene synthesis), which ligates, or glues, short single-stranded DNA (ssDNA) together to chemically synthesize genes. Although this general process can be carried out by many different procedures in lab, column-based (or column enclosed) oligonucleotide synthesis remains as an efficient and cost-effective method by which DNA is synthesized. Through a “four-step chain elongation cycle,” ssDNA ranging from 60 to 100 nucleotides are chemically synthesized. In their order, these steps include deprotection, coupling, oxidation, and capping. After these short molecules have been synthesized, nucleotides in each molecule are aligned through a specifically designed base pairing process. An enzyme commonly used in cellular DNA synthesis called DNA Polymerase is used to both fill-in any gaps in the molecule created during the base pairing process and proofread the sequence. The product is a synthetic gene sequence that is identical to one that would be synthesized through a cellular process; the difference, however, is that laboratory researchers can control this synthetic sequence. With current technology, this method can synthesize individual ssDNA at “scales ranging from 10 to 1000 nmol” with costs anywhere between “$0.05 and $0.17 per base” (Hughes). Because of these constraints on synthesis, scaling with larger sequences is a challenge.

RATIONAL:

Stemming from the need to decrease the cost of synthetic synthesis, the rationale behind this research begins with the introduction of a hypothesized catabolic synthesis method (referred to as restriction synthesis): a process that breaks down a DNA into fragments using many endonucleases in order to synthesize a gene. This process is an extension of the BioBricks™ standard proposed by Dr. Thomas Knight. In their publication, BioBricks™ is defined as a process that “employs iterative restriction enzyme digestion and ligation reactions to assemble small basic parts into larger composite parts.” These larger components are built using fragments flanked by two restriction enzymes, Xbal and SpeI (Anderson). One key difference between restriction synthesis and BioBricks™ is that the latter restricts digestion to those two enzymes: XbaI and SpeI. Conversely, restriction synthesis relies on a dataset of more than 200 enzymes all with unique restriction cutting sites. In flanking subsequences with a greater number of available restriction enzymes, restriction synthesis was originally thought to build many more composite subsequences, which are homologous to those referenced by the BioBricks™ standard.



The development of an algorithmic simulation of restriction synthesis began in 2016. Since the project’s inception, the estimated cost of synthesis has shown to decrease rapidly as the simulation grew more representative of the proposed lab procedure. With costs now as low as $0.019/bp, this simulation is ready for its next stage of development, predictive subsequence classification, which will be major topic of this paper.

In regard to synthesizing DNA, catabolic synthesis is counterintuitive compared to the anabolic methods. Nevertheless, this process generates low simulated costs because of its use of restriction enzymes. During the simulation of synthesis, the algorithm searches for applicable sequences. As it will be explained more in depth later in the report, an applicable sequence is one that can be flanked and digested by two enzymes and be end-to-end compatible with sticky-ended sequences before and after it. Therefore, the total cost of synthesizing a fragment is determined by the individual unit cost of the enzymes used, and the cost per base pair ($/bp) is determined by the length of the flanked sequence. In this way, longer sequences spread the overhead of the cost for each iteration, decreasing the total for the entire process. In other conventional methods such as anabolic gene synthesis, the cost of production is fixed to the sum of each singular reaction of nucleotide-to-nucleotide elongation. As the main determinant of production, this cost metric is integral to increasing the efficiency of the gene synthesis process.

ALGORITHM:

Restriction synthesis is currently modeled computational based on an iterative restriction enzyme digest and sticky end alignment process. Given a gene and a larger DNA sequence (also known as the set), the algorithm simulates the process by first searching for fragments (or subsets) in the set that sequentially appear in the gene. These matches are considered subsequence matches, which is one of three conditions required for a fragment to be used for the synthesis process. The length of these fragments varies with each search depending on what subsets are available in the set. It is more likely, for instance, to find a subsequence match of length seven compared to a subsequence match of length fourteen. Taking this into account, each subsequence search begins with parsing for subsequences of length t, and then continues by decrease this value until a subsequence match is found (t is set to 16 by default). Through this method, each iteration for a subsequence match ensures that the longest possible subsequence match is found.

An instance match, unlike a subsequence match, is based on whether two enzymes exist that can flank and digest the fragment in the set. This condition is critical because while there exist thousands of subsequence matches, only a select few may be applicable for the synthesis process—in order for a subsequence to be included from the set, it needs to be removed by flanked enzymatic digest (Anderson). Using the New England Biolabs’ restriction enzyme database, the algorithm has access to 204 unique restriction enzymes and their respective restriction sites (New England Biolabs). Despite this low number of enzymes, many have versatile recognition sequences, cutting a variety of DNA patterns. Access to these versatile enzymes is one of the reasons why this process can successfully complete on a variety of genes.

In addition to a subsequence match and an instance match, a third match is required: a ligation match. Some restriction enzymes cut DNA in a Z-like manner, where the product of the enzymatic digest contains nucleotide overhangs, commonly referred to as sticky ends. Because many of the enzymes create these sticky ends on digested fragments, a ligation match is needed to ensure that sequential fragments are compatible. For instance, a subsequence that ends in an overhang of AGTA on the 5’ end needs to be followed with an overhang that begins with TCAT on the 3’ end. Also, fragments cut with enzymes that produce blunt ends, or ends without any overhangs, need to be followed with fragments that have these produced blunt ends as well. This final end-to-end compatibility indicates whether a fragment is complete matche and whether the search process continues.

These three main conditions, subsequence match, instance match, and ligation match, dictate whether a subset in the set can be including in the synthesis process. Specifically, the two former conditions rely entirely on the sequence of the gene and the set. For this reason, both DNA sequences need to be known prior to performing this method. This assumption would require costly sequencing if each gene was synthesized from a different set. In other words, each unique gene sequence would have a respective sequenced set from which subsequences would be pulled. While this solution would allow for incredibly accurate synthesis, it would massively increase the cost of this method. Another solution is to have one singular set that can be applied to a category of genes, ones that are alike in their general nucleotide composition. While this solution would allow for prices to remain constantly low, it requires choosing which subsets of fragments to include a generalized set. This proposed solution is the most cost-effective out of the two. For this reason, the question of whether to include a subset in the set was explored in the research.

With the use of restriction synthesis data obtained from mycobacterium structural genes (ranging from 1000 to 5000 bp) collected from the National Center for Biotechnology Information, the analysis of different gene sequences was explored using three popular techniques in machine learning and neural network architecture: Support Vector Machines (SVMs), Random Forests, and Convolution Neural Networks (CNNs) (Nucleotide [Internet]).

**DATA**

FEATURES:

The feature set includes 16 unique variables about each entry in the data set. The first one, ‘seq,’ is the most unique feature as it depends on the subsequences in the data set itself. Because the goal of this network is to determine whether to include a subsequence in a set, the individual nucleotides and their positions need to be inspected as individual features. All in all this relates to whether the subsequence can undergo flanked enzymatic digest.

Other features that need explanations are the different complexity ratings of the subsequence and of the gene. These are used to accurately describe the nucleotide composition with a single value from 0 exclusive to 1 inclusive. The equations for the calculations are below Figure 1. Along with the nucleotide proportion features for the subsequences, these complexity ratings determine how “complex” a strand is. For instance, the two extremes are a subsequence with only one repeating nucleotide in its sequence and one with equal nucleotide composition in its sequence. The former has a rating closer to 0 as it is entirely uniform and the latter has a rating of 1 as there is an equal distribution of each nucleotide, making it more complex.

**Figure 1:** Feature Set

|  |  |
| --- | --- |
| Feature | Description |
| seq | This feature contains as many sub-features as there are nucleotides in the longest sequence with the following assignment: seq(‘N’) = 0, seq(‘A’) = 1, seq(‘T’) = 2, seq(‘C’) = 3, seq(‘G’) = 4. |
| len | Length of the subsequence. |
| pA | Local proportion of A’s in the subsequence. |
| pT | Local proportion of T’s in the subsequence. |
| pC | Local proportion of C’s in the subsequence. |
| pG | Local proportion of G’s in the subsequence. |
| sr1m | Local complete complexity rating of the subsequence. |
| sr2m | Local proportional complexity rating subsequence. |
| C | Global proportion of genes that contain the subsequence. |
| gpA | Local proportion of A’s in the gene. |
| gpT | Local proportion of T’s in the gene. |
| gpC | Local proportion of C’s in the gene. |
| gpG | Local proportion of G’s in the gene. |
| gR1n | Local complexing rating of the gene. |
| gR2n | Local proportional complexity rating of the gene. |
| en | Local number of available enzymes that digest the subsequence. |

Note: The word “local” pertains to the immediate gene and “global” pertains to the set of genes under study, and m is the length of the subsequence and n is the length of the gene.

DATA SELECTION:

The entries in the data set were determined based on simulations of restriction synthesis. For instance, during the synthesis of a gene, all subsequences were analyzed and categorized based on whether they were applicable for synthesis. Only subsequences that resulted in a subsequence match were considered, and subsequences that then resulted in both an instance match and a ligation match were given a Boolean true value. Subsequences that did not satisfy both of those two conditions were given a Boolean false value. Intuitively, since there is a finite number of enzymes and a finite number of ways to generate a ligation match between subsequences, one would believe that would exist many more false entries than true entries; this was observed during data collection. As a proportional distribution would be most effective for machine learning, this was not ideal. A solution to this issue will be explained in the next section.

Moreover, duplicate subsequence entries between genes were not removed because six of the features are gene sequence dependent—those features use characteristics from the sequence of the gene and not the individual subsequence in question. Therefore, just because two subsequences matched did not indicate that they had the same feature set. This greatly reduced the complexity of the algorithm, allowing for many entries to be generated fairly quickly. Duplicates subsequence entries within a gene were also not removed. Due to the small size of each gene and the subsequence’s reliance of ligation matches, these duplicates were considered negligible.

DATA CURATION:

Throughout the research, a constant training set and a constant test set were used while evaluating each of the machine learning solutions. Based on an initial raw data set of approximately one million entries across almost one hundred gene synthesis simulations, a sample of 105,000 entries were selected with 50/50 percent distribution of true and false output values. From that sample, a random sample produced both the training set and test set with unique entries. The training set consisted of 50,000 unique entries with a 60/40 percent distribution of false and true outputs respectively, and the test set consisted of 26,623 unique entries with a 50/50 percent distribution of false and true outputs respectively.

This data curation along with other data analysis were carried out with several MATLAB functions built to efficiently separate and analyze the data. These functions along with their documentation are included on the project’s GitHub repositiory.

FEATURE ANALYSIS:

Prior to running all of the machine learning techniques on the data, a correlation test was applied across all features to determine whether any were redundant. The threshold for redundancy with respect to correlation was set at 0.90 because the data set already had far less features than were optimal. Nonetheless, there were no features with correlations higher than 0.80, so none of them were removed from the feature set.

**Figure 2:** Correlation Test between Features

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  | nuc\_1 | nuc\_2 | nuc\_3 | nuc\_4 | nuc\_5 | nuc\_6 |
| nuc\_1 | 1 |  |  |  |  |  |
| nuc\_2 | 0.05075186 | 1 |  |  |  |  |
| nuc\_3 | -0.0129095 | 0.00637293 | 1 |  |  |  |
| nuc\_4 | -0.062499 | -0.0071364 | -0.0023504 | 1 |  |  |
| nuc\_5 | -0.0153083 | 0.08837091 | -0.0116218 | 0.00100385 | 1 |  |
| nuc\_6 | -0.0418458 | 0.02542745 | 0.11868449 | -0.0165755 | -0.0276393 | 1 |

Additionally, at this stage of the research feature reduction was approached with reluctance as there were not many selected features to begin with. As the research continues and more features are selected, feature reduction may be relied on more heavily as a part of the preprocessing of the data. Despite this fact, feature reduction was performed in order to ensure the data’s effectiveness between each of the three machine learning techniques.

**MEHTODS**

SVMs

1. Background

Support Vector Machines (SVMs) are supervised learning models that create hyperplanes between groups of data. Given a set of training data and corresponding classifications, the algorithm attempts to best categorize data by measuring each data point’s distance to the segregating hyperplane. By maximizing these distances, SVMs create highly reliable classifications. Commonly used for classification, regression, and outlier detection, this algorithm is in it of itself a non-probabilistic classifier (Scikit-Learn SVM). Because the researcher cared more about definitive classification and not probabilistic classification, this technique was applicable to both the research problem and the data set.

While SVMs are powerful, they still require certain input parameters from the researcher that help to produce optimal classification results, such as the regularization parameter, class weight, kernel coefficient, etc. Although some data sets may not be linearly separable, kernel functions, such as polymetric kernel, gaussian kernel, and sigmoid kernel, may be used to categorize different these types of data sets (Support Vector Machines, 2). In addition to SVMs’ versatility on different data classifications patterns, they are highly efficient in higher dimension feature space and when the number of samples is less than the dimension of the feature space (Scikit-Learn SVM). In the case of this research, this higher dimension feature space efficiency was one of the main reasons SVMs was used.

In addition to using the base 16 selected features, Principle Component Analysis (PCA) was applied to reduce the feature space.

1. Results

**Figure 3:** Polymetric SVM Applied With PCA

A close up of a map

Description automatically generated

As seen in Figure 2, when 2-component PCA was used the classification can be visualized. In this case, the most successful classifier was the polymetric, 2-component PCA. The false classification accuracy was 81%, and the true classification accuracy was 94%. These results were seen consistently across all trials of 2-compondent PCA. For this reason, the standard deviation is considered negligible.

**Figure 4:** Confusion Matrix for SVM Analysis

|  |  |  |
| --- | --- | --- |
|  | Actual True | Actual False |
| Predicted True | 12632 | 678 |
| Predicted False | 3009 | 10304 |

In Figure 4, the confusion matrix displays the distribution of false positives and false negatives which were \_\_% and \_\_% respectively.

**Figure 5:** Uniform and Nonuniform Feature Distribution for SVM Analysis

|  |  |
| --- | --- |
| A screenshot of a cell phone  Description automatically generated | A picture containing text  Description automatically generated |

These histograms display the relative proportions of each nucleotide at different features. Across all 46 single nucleotide features (in the case of this data set), many of them had distributions similar to the two above. Wehn the distribution is seemingly uniform, there is a displayed indifference between the four nucleotides. This could suggest that at least across this data set, the eighth nucleotide position was insensitive to the type of base. Conversely, the figure on the right displays higher base sensitivity as it favors cytosine far higher than any of the other three nucleotides. This distribution suggests the first position is incredibly sensitive to the type of base.

Further analysis is needed in order to determine whether these differences are statistically significant and whether they would further add to the study of building a set for restriction synthesis. As it currently stands, this pseudo-quantitative analysis suggests a difference between the two positions in their nucleotide sensitivity.

Upon further analysis of these histograms, one can determine why at certain positions there were misclassifications. For instance, the left histogram in Figure 5 shows both over and under classification of true and false sequences. Specifically, the “nuc\_8” feature misclassified all four bases in equal proportions. In other words, further statistical analysis may suggest that there is not a significant difference between misclassification rates for each nucleotide. Additionally, the right histogram displays that the “nuc\_1” feature misclassified cytosine and guanine in much higher proportions than adenosine and thymine. Further statically analysis may suggest this to be the case. More interestingly, these misclassifications seem to match the actual distributions of correctly matched sequences. For example, the left histogram had misclassified sequences uniformly across that feature, which follows the distribution of the true classification at that position. This phenomenon is also apparent in the right histogram as the proportion of misclassifications follow the distribution of true classifications at that feature.

1. Alternatives

In addition to this PCA polymetric SVMs, five other tests were conducted: non-PCA linear kernel; 3-component PCA linear kernel; 3-component PCA polymetric kernel; 2-component PCA RBF kernel; and 2-component PCA sigmoid kernel. The non-PCA linear kernel displayed an 84% false accuracy and an 89% true accuracy; the 3-component PCA linear kernel displayed an 81% false accuracy and a 92% true accuracy; the 3-component PCA polymetric kernel displayed an 82% false accuracy and a 93% false accuracy; the 2-component PCA RBF kernel displayed an 82% false accuracy and a 93% true accuracy; and the 2-component PCA sigmoid kernel displayed a 76% false accuracy and a 75% true accuracy. While the 3-component PCA polymetric kernel and the 2-component PCA RBF kernel are arguably better than the other three, they do not display high enough true categorization accuracies in ordered to be the favored SVMs model.

**Figure 6:** 2-Component PCA RBF Kernel (left) and 2-Component PCA Sigmoid Kernel (right)

|  |  |
| --- | --- |
| A picture containing map  Description automatically generated | A close up of a map  Description automatically generated |

While other kernels were applied, such as polymetric, RBF, and sigmoid, to non-PCA SVMs analysis, the training set and test set were nonconstant, so their results need not be critically inspected. All three displayed higher false accuracy classification than true accuracy classification through.

RANDOM FORESTS

1. Background

Random forests are an ensemble learning model that implement many decision trees based on subsets of training data for classification and regression. One of the features of random forest is its use of out-of-bag data. At the constriction of each tree, two-thirds of the data is included for training and the remaining one-third is set aside for validation after the training has completed (Breiman). During the training process, each decision tree is generated based on a bootstrap data set and a randomly selected feature set at each note of the tree (Cutler). While a single decision tree would overfit data, random forests’ use of many decision trees allows the overfitting to average out between trees (Scikit-Learn Random Forest). When predicting based on new data, each tree provides a predication, or a vote, for a certain class. The classification for random forest is based on which classification has the most votes.

Some of the advantages of the random forest model include a high accuracy across data sets, the ability to process many features regardless if they contribute greatly to the output, and its insensitivity to outliers, noisy or missing data, and different variable types (Devetyarov). While many of these advantages do not directly concern this classification problem, random forest does use bagging in order to reduce overfitting. Because this is new data, this problem was one of the researcher’s main concerns.

1. Results

**Figure 7:** Confusion Matrix for Random Forest (n=50) Analysis

|  |  |  |
| --- | --- | --- |
|  | Actual True | Actual False |
| Predicted True | 12336 | 974 |
| Predicted False | 1898 | 11415 |

The false classification accuracy was 87%, and the true classification accuracy was 92%. These results were seen consistently across all trials of random forest. For this reason, the standard deviation is considered negligible.

1. Alternatives

The random forest analysis was tested with varying amounts of trees, or values of n: 10, 20, 30, 40, and 50. The most successful analysis was conducted when n was 50, but even as n passed 40, the additional level of accuracy in both categorizations began to level off. For example, the classification accuracy when n was 50 was the same when n was 30.

CONVOLUTION NEURAL NETWORKS

1. Background

Convolution Neural Networks (CNNs) are a class of deep learning algorithms that use a linear operation called convolution in at least one of their layers (Goodfellow). In CNNs, the model always begins with a convolution layer that accepts a tensor as an input with dimensions based on the size of the data. The second layer, or the first hidden layer, is formed by applying a kernel or filter that is a smaller matrix of weights over a receptive field, a small subspace of the inputs. These kernels apply an inner product on the receptive field effectively compressing the size of the input space. As the kernel strides across the inputs, the first hidden layer is computed based on the weights of the filter. As a result, the first hidden layer is a feature map formed from the kernel applied on the input space. While the dimension of the kernel may be much smaller in size compared to the initial inputs of the convolution layer, the kernel must have the same depth of the input space. The inputs and convolution layer are often followed by rounds of activations, normalizations, and pooling. The last layer, however, is a fully connected layer where the final outputs and categorizations are determined based on how different features fall in line with the specific classes under study (Deshpande).

1. Results

**Figure 8:** Currently Optimal Convolution Neural Network

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| # | Layer Type | Parameters | Output | # of Params |
| 1 | Conv1D | Kernel size = 3, softplus activation | (1, 64) | 11776 |
| 2 | Conv1D | Kernel size = 3, linear activation, padding = same, strides = 3, Kernel regularizer, Activity regularizer | (1, 64) | 12352 |
| 3 | Droupout | Rate = 0.5 | (1, 64) | 0 |
| 4 | MaxPooling1D | Pool size = 2, padding = same | (1, 64) | 0 |
| 5 | Flatten | None | (1, 64) | 0 |
| 6 | Dense | Linear activation | (1, 128) | 8320 |
| 7 | Dense | Softmax activation | (1, 2) | 258 |

Total params: 32,706

Trainable params: 32,706

Non-trainable params: 0

The false classification accuracy was 79.7% with a standard deviation of 3.4%, and the true classification accuracy was 93.0% with a standard deviation of 3.1%, but the.

1. Alternatives

Originally, the CNNs analysis began with a model analogous to a two-dimensional image convolution network. This included three rounds of convolution, rectified linear unit activation, and maximum pooling. Ending with a fully connected dense layer, the true classification was around 88% and determined to be too low for the finished model because it was less than both SVMs and random forest. It should be noted that this classification accuracy was incredibly consistent from trial to trial. The addition of bath normalization produced a true accuracy upwards of 95%, but the cross validation displayed true accuracy closer to 80%, which suggest that this layer simply produced overfitting. In other words, the results were unable to be replicated on new data.

The next model under examination was used from missinglink.aibuilt specifically for 1-dimensional inputs (“Keras Conv1d…”). This design alone produced a classification accuracy was 81.0% with a standard deviation of 2.5%, and the true classification accuracy was 91.2% with a standard deviation of 3.03%. An extension of the model was made with both a Kernel regularizer and Activity regularizer. This led to the final classification accuracies. Further consideration is needed to increase both the true and false classification accuracy.

**DISSCUSION**

FALSE POSITIVES/FALSE NEGATIVES

When looking at these model, one post hoc inquiry might be whether false positives or false negatives are less desirable than the other. In the scope of this research, a false positive is deciding to include a subsequence in the set when it should not have been included, and a false negative is deciding not to include a subsequence in the set when it should have been. In all of the models, there were far more false positives than false negatives. Initially, one would think that false positives should be avoided more than false negatives. One of the arguments for this proposition is that fewer false subsequences that do not add to the set would allow the set to be smaller in size. Since the size of the set was not determined to be a major constraint on the set, this need not be an issue. Moreover, a further consideration is that the length of the genes under question are only 1000 to 5000 bp long and DNA is notoriously long in it of itself. This further explains why the size of the set is not of the most importance at this point in the research.

Instead, the inclusion of these false positives increases the genetic variation in the set if it is used recursively. A large set would be generated from entries of a randomly generated DNA sequence. This set would be then subsequently used in determining more entries and so on and so forth. Genetic variation is paramount to the field of genomics, and while many variation techniques exist in machine learning already, the recursive use of sets allows for the existence of more subsequences to be taken into account with new models and techniques.

COMPARISON

The differences between these three algorithms pose an interesting discussion. While SVMs and random forest are far more consistent than CNNs, they both had a lower true classifications average accuracy compared to that of CNNs—that of random forest and of SVMs was 92% and 94% respectively. Furthermore, the average false classification accuracy between all three vary greatly with random forest performing better than SVMs and CNNs. Of all three, CNNs had the lowest false classification accuracy at \_\_%, while SVMs trailed behind random forest by only 6%. In then comparing SVMs to random forest based on the true classification accuracy, the question becomes whether a 2% increase in the true classification accuracy is worth a 6% decrease in false classification accuracy. Because of the false positive and false negative discussion in the previous section, the answer is definitely yes. For this reason, SVMs is ranked ahead of random forest in these respects.

Because of the consistency mentioned earlier, the comparison of SVMs and random forest with CNNs is not as clear. While an increase in false positives may not be terrible at this stage of the research, consistency in the algorithms performance always matter because of the recursive nature of the hypothesized set selection process.

On the topic of the observed classification accuracies, all three models exhibited low accuracies with respect to the false classification in particular. The highest observed classification accuracy was only 87% with random forest. False classification accuracies above 75% are considered acceptable because of the recursive nature of the proposed set selection processes. It is interesting, though, that all three techniques exhibited low false classification accuracies. One possible interpretation is that true cases are easier to differentiate from false cases than false cases are from true cases.

CONCLUSION

All in all, SVMs performed far better in true classification compared to CNNs and Random Forest. One hypothesis for why this might have been the case is due to its ability to build components through PCA. Surprisingly, all 16 unique features were able to be reduced to only two components, allowing for the classification to be clearly evident in two-dimensional space. Although the 95% benchmark for true classification accuracy was not reached with any of the three machine learning techniques, all three did surpass 90%. This indicates that better classification is possible, but it may require more work in the realm of feature selection and data preprocessing.

FUTURE RESEARCH

As it stands, the SVMs analysis does an adequate job of recognizing true classifications from the gene data under question. This classification is limited because of the high rate of false positives. As previously mentioned, this is ideal for initially generating sets with artificial genetic variation. In the future, it would be fundamental to this research to build and explore models that are more conservative. Models such of these would be robust in deciding whether randomly generated sequences from a random gene should be included in a set. The models explored in this study cannot accomplish categorization in that respect.

Despite the lower relative performance of the CNNs, future models with this analysis may produce better results than what were observed here. One of the reasons why this was the case is most likely that the feature set was not explored entirely. If the data was not explained in its entirety, CNNs might have lacked the available information to correctly classify which sequences should be included in the set. With this in mind, the feature set needs to be explored more in depth so CNNs can more effective classify the data.

**REFERENCES**

Anderson, J Christopher et al. “BglBricks: A flexible standard for biological part assembly.” Journal of biological engineering vol. 4,1 1. 20 Jan. 2010, doi:10.1186/1754-1611-4-1. Accessed Web. 24 Oct 2019.

#### Breiman L., Cutler A."Random Forests - Classification Description." *Stat.berkeley.edu*. Accessed Web. 28 Feb. 2020.

Cooper, Geoffrey. "DNA Replication." *Sinauer Associates* (2000). Accessed Web. 27 Feb. 2020.

"CRISPR Timeline." *Broad Institute*. 2015. Accessed Web. 3 Mar. 2020.

Cutler A., and J. R. Stevens. 2006. Random forests for microarrays. Methods in Enzymology 411:422-432. Accessed Web. 28 February 2020.

Deshpande, Adit. "A Beginner's Guide To Understanding Convolutional Neural Networks." Adeshpande3.github. 2016. Accessed Web. 29 Feb. 2020.

Devetyarov D., Nouretdinov I. (2010) Prediction with Confidence Based on a Random Forest Classifier. In: Papadopoulos H., Andreou A.S., Bramer M. (eds) Artificial Intelligence Applications and Innovations. AIAI 2010. IFIP Advances in Information and Communication Technology, vol 339. Springer, Berlin, Heidelberg. Accessed Web. 28 February 2020.

Goodfellow, Ian, et al. Deep Learning. The MIT Press, 2017. p. 326

Hughes, Randall A, and Andrew D Ellington. “Synthetic DNA Synthesis and Assembly: Putting the Synthetic in Synthetic Biology.” Cold Spring Harbor perspectives in biology vol. 9,1 a023812. 3 Jan. 2017, doi:10.1101/cshperspect.a023812. Accessed Web. 24 Oct 2019.

"Keras Conv1d: Working With 1D Convolutional Neural Networks In Keras - Missinglink.Ai." *MissingLink.ai.* Accessed Web. 2 Mar. 2020.

New England Biolabs. “Restriction Endonucleases.” NEB, www.neb.com/products/restriction-endonucleases. Accessed Web. 29 Oct 2019.

Nucleotide [Internet]. Bethesda (MD): National Library of Medicine (US), National Center for Biotechnology Information; 2004 – Accessed Web. 24 Jan. 2020. Available from: https://www.ncbi.nlm.nih.gov/nuccore/

Support Vector Machines: Data Analysis, Machine Learning and Applications, edited by Brandon H. Boyle, Nova Science Publishers, Incorporated, 2011. ProQuest Ebook Central, https://ebookcentral-proquest-com.ezproxy2.library.drexel.edu/lib/drexel-ebooks/detail.action?docID=3021500. Accessed Web. 27 February 2020.

#### Scikit-Learn SVM. "1.4. Support Vector Machines — Scikit-Learn 0.22.1 Documentation." *Scikit-learn.org*. Accessed Web. 27 Feb. 2020.

Scikit-Learn Random Forest."3.2.4.3.1. Sklearn.Ensemble.Randomforestclassifier — Scikit-Learn 0.22.1 Documentation." Scikit-learn.org. Accessed Web. 28 Feb. 2020.

Tang, Nicholas, Siying Ma, and Jingdong Tian. "New Tools For Cost-Effective DNA Synthesis." Synthetic Biology (2013): 3-21. Accessed Web. 1 Mar. 2020.

"The Human Genome Project." Genome.gov. *National Human Genome Research Institute*. 2020. Accessed Web. 3 Mar. 2020.