VERTICALLY INTEGRATED

PROJECT (VIP) REPORT

NEUROMORPHIC SOLUTIONS TO HGIH DIMENSIONAL DNA SUBSEQUENCE MATCHING VIA ALIGNMENT DATA

*Ethan Jacob Moyer*

ENGR 370: Applications, Algorithms, & Architecture for

Neuromorphic Computing

*Dr. Anup Kumar Das*

Drexel University

4 May 2020

**INTRODUCTION**

BACKROUND:

[Introduction of alignment]

Sequence alignment of biological molecules, such as deoxyribonucleic acid (DNA), focuses on identifying regions that are of high similarity due to common factors. Typically, these factors are derived from functional, structural, or evolutionary relationships shared between the sequences being aligned [1]. More specifically, the field of bioinformatics is concerned with sequence alignment algorithms as longer sequences become increasingly difficult to align by hand. One such algorithm is Pairwise Sequence Alignment (PSA). In this algorithm, each element in a query sequence (the sequence being aligned) is compared each element of a reference sequence (the sequence to which the query is being aligned). Although it has mainly two different dynamic programming implementations, such as through the Needle-Wunsch (NW) [2] and Smith-Waterman (SW) [3] techniques, the general theme of alignment remains constant. A positive score is assigned to matching elements and a negative score is assigned to nonmatching elements. In this way, the optimal alignment between a query and a reference is identified by the highest scoring segment [4]. Different scoring matrixes can be used to alter these scores. For instance, …

[Application of alignment to this research]

The main focus of this research only involves the nucleotide similarity between sequences. As a result, the project is structured around identifying sequence matches rather than sequence alignments, which as previously mentioned requires evolutionary data. The scope of this research operationally defines subsequence matching as the matching of k-mer sequences based entirely on their percent similarity with respect to PSA scores. Since the SW technique focuses primarily on local alignment while the NW technique focuses on global alignment, the former will be used as the basis for alignment scores.

Each k-mer is matched based on a 95 percent similarity threshold; more simply, if when aligning with respect to a reference k-mer, a query k-mer scores equal to or higher than 95 percent of the score when the reference is aligned to itself, then it is considered to be matched.

RATIONAL:

Originally, the premise behind this research was to determine where a match exists in a sequence of k-mers. This was thought to correspond to the position of a match, which would serve as an intrinsic measurement for the matching of two k-mers. Since each match is based on a 95 percent similarity threshold, these k-mers can be classified into classes. Analogous to equivalence classes, which in mathematics are similar sets when an equivalence relation is imposed on them, these classes serve as groups that house k-mers that are at least 95% similar.

[4] Salamat, S. and Rosing, T., 2020. FPGA Acceleration Of Sequence Alignment: A Survey. UC San Diego: Department of Computer Science and Engineering, pp.2,3.

[1] Mount DM. (2004). Bioinformatics: Sequence and Genome Analysis (2nd ed.). Cold Spring Harbor Laboratory Press: Cold Spring Harbor, NY.

[2] S. B. Needleman and C. D. Wunsch, “A general method applicable to the search for similarities in the amino acid sequence of two proteins,” Journal of molecular biology, vol. 48, no. 3, pp. 443–453, 1970.

[3] T. F. Smith, M. S. Waterman et al., “Identification of common molecular subsequences,” Journal of molecular biology, vol. 147, no. 1, pp. 195–197, 1981.

ANALOGY:

An analogous problem is asking to classify similar words across repeats of the same sentence. This might be based on high dimensional quantifiers, such as properties of being a noun, verb, conjunction, etc., or context indicators that the English language takes for granted, such as verb-noun agreement, implied antecedents, etc. Using context cues, a computer could generate disjoint data for different words. For instance, one sentence copy would be classified solely based on a specific class of closely related nouns, such as the noun ‘CAR.’ An algorithm might place different subtypes of cars into that word class, like ‘TRUCK’ or ‘VAN.’ A separate sentence copy would be classified based on another English quantifier or context indicator. In this way, sentence data would be organized into discrete classes, each containing disjoint information about its respective word group. As one could imagine, the task of classifying words is trivial for short sentences and when there are relatively few classes. The task becomes more intensive once both of these attributes increase.

Such an increase is observed when applying this problem structure to DNA subsequence matching. It is similar to word classification problem for spoken languages in that both problems are based on text classification. One difference between the two is the convention by which text is interpreted. In English, sentences are composed of words, each of which is comprised of an arrangement of 26 unique letters. In the genetic language, these sequences are composed of a composition of four unique bases. This distinction between *an arrangement* in a language like English and *a composition* in the genetic language is imperative of the increased complexity when applying word classification to DNA subsequence classification.

An arrangement is operationally defined as grouping of characters where only specific permutations carry meaning. For example, the word ‘ACR’ has little meaning compared to ‘CAR’ and ‘ARC.’ A composition is operationally defined as a grouping of characters where generally all permutations may carry meaning. In the case of DNA nucleotides, ‘ATCAGTC’ caries as much meaning as ‘TTCCAA.’ As displayed by the provided examples, the concept of an arrangement pairs nicely with English words and the concept of a composition pairs nicely with the genetic language. In most cases, the length of subsequences is much greater than the length of words. As previously mentioned, it becomes increasingly difficult to classify subsequences as the length of each subsequence and the number of possible classes increases. For these reasons, DNA subsequence matching is much more complex than language word classification; nonetheless, high dimensional word classification serves as a perfect analogy from DNA subsequence matching can be built.

Moreover, the need for the neural network in the analogous word classifying problem is strikingly similar to our problem in DNA subsequence matching. If the number of quantifiers is low, there really isn’t much need for a network to do prediction. However, as these quantifiers increases increase, the complexity of classifying words (or in our case, subsequences) increases exponentially. Similar to how my MATLAB-produced files containing disjoint information about different subsequence matches, so too could a ‘dictionary’ file contain disjoint information about words in different contexts for a variety of quantifiers.

One aspect that the word classification model does not directly address is the ability to determine (and predict) where matches occur in a sentence. This is of course more critical in DNA subsequence matching as the position of subsequence matches, as opposed to the quality of having a match, is of much greater importance. Regardless, this will be achieved through careful curation of the data and the use of Long Short-Term Memory (LSTM) Recurrent Neural Networks (RNNs).

**DATA**

STRUCTURE:

In order to use machine learning to predict protein alignment, it was postulated that there needs to be two main aspects of the data and model. The first hypothesized aspect was that the data needs to somehow resemble alignment of sequences. It is far too difficult to generate alignment scores or positions based off a singular sequence. To simply the research, the main output focus was the positions of the most optimal alignment instead of alignment scores. In order to resemble optimal alignment positions in the data, the sequences were split up into sequentially appearing k-mer segments, or subsequences of k long molecules, each one starting one more nucleotide to the right than the preceding one. The position of each subsequent k-mer in a list represents the position at which it appears in the sequence. Structured in a table, the first column is this list of k-mer sequences. The second column is a logical list identifying where a selected sequence appears in the list of k-mer sequences. In this way, the position of logical true values in the list identify where an alignment occurs in the sequence because of the one-to-one correspondence between the logical value list and the list of k-mer subsequences.

The following figure illustrates the basic set up of the data structure that was fed through the LSTM model

**Figure 1:** Entry Data Structure

Query (Sq)

|  |  |
| --- | --- |
| Reference (Sr) | Logical (L) |
| k0 | m0 |
| k1 | m1 |
| . | . |
| . | . |
| . | . |
| ki | mi |
| . | . |
| . | . |
| . | . |
| kn-1 | mn-1 |
| kn | mn |

where ki represents an individual k-mer segment of the reference sequence Sr and mi represents the logical value of a match of the query sequence Sq against ki.

COLLECTION:

The DNA sequence that was generated to create the training, test, and validation data sets was obtained using the *randseq* function from the MATLAB Bioinformatics Toolbox. In a custom MATLAB function, *generateDNA*, *randseq* generates a sequence of length n and saves it to a specified file, ‘dna.txt.’ In another custom function, *gatherData*, a cell character array of short k-mers are generated for a certain small length. These k-mers are then loop through in sequence and fed into another custom function, *generateNonopTable*, with the name of the sequence stored in ‘dna.txt,’ the cell character array of k-mers, the length of the generated k-mers, and an alpha value that corresponds to the 95% percent similarity in the local alignment matches between sequence. In the *generateNonopTable* function, a table of sequence matches for each k-mer given to the function is generated by aligning the given k-mer with the sequence of k-mers at 95% similarity. Specifically, through the use of the MATLAB Bioinformatics Toolbox, a modified (SW) alignment technique is used through the function *localalign* [1] in order to perform the alignments of each k-mer in *generateNonopTable*. The result of each iteration is a structure, which is represented in Figure 1, of k-mers and their respective Boolean matching value when compared to k-mers already existing in the sequence. In this way, only data that is applicable to the sequence is generated and included in the model data sets. Duplicate data is removed automatically.

[1] Barton, G. (1993). An efficient algorithm to locate all locally optimal alignments between two sequences allowing for gaps. CABIOS 9, 729–734.

PIPELINE:

The following diagram displays the data pipeline for the paradigm of how the model should function

**Figure 2:** Data Pipeline

S0, S1, … , Sj, … , Sn-1, Sn

Model trained on specific set at 95% similarity

L0, L1, … , Lj, … , Ln-1, Ln

where Sj represents a list of k-mer fragments of a unique gene sequence and Lj represents a list of logical values corresponding to the occurrences of those k-mer fragments within the specific set.

CURATION:

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 repository.

**MODEL**

PROPOSED MODEL:

<https://github.com/philippmuench/dna_lstm>

The structure of the data leads into the second hypothesized aspect: the use of a long short-term memory (LSTM) recurrent neural network (RNN) based model. Since the list of k-mer subsequences is structured like time series, LSTM is arguably the best model for this format of data.

**Figure 3:** Machine Learning Model

Input Embedding Layer

Output Dense Layer

Time Distributed LSTM Layer

Dropout Layer

Dropout Layer

Time Distributed LSTM Layer

**MEHTODS**

LSTM RNN

1. Background

LSTM neural networks [2] overcome the vanishing gradient problem experienced by recurrent neural networks (RNNs) by employing multiplicative gates that enforce constant error flow through the internal states of special units called ‘memory cells’. The input (IG), output (OG), and forget (FG) gates prevent memory contents from being perturbed by irrelevant inputs and outputs (refer Fig. 1(a)), thereby allowing for long term memory storage. Because of this ability to learn long term correlations in a sequence, LSTM networks obviate the need for a pre-specified time window and are capable of accurately modelling complex multivariate sequences. In this paper, we demonstrate that by modelling the normal behaviour of a time series via stacked LSTM networks, we can accurately detect deviations from normal behaviour without any pre-specified context window or preprocessing.

Long Short Term Memory (LSTM) networks have been demonstrated to be particularly useful for learning sequences containing longer term patterns of unknown length, due to their ability to maintain long term memory. Stacking recurrent hidden layers in such networks also enables the learning of higher level temporal features, for faster learning with sparser representations

Verleysen, Michel. Proceedings / 23Rd European Symposium On Artificial Neural Networks, Computational Intelligence And Machine Learning, ESANN 2015, Bruges, Belgium, April 22-23-24, 2015. Ciaco, 2015.

1. Results

Despite the fact that the DNA data was set up as time steps, the LSTM failed to correctly identify new true cases.

1. Alternatives

**DISSCUSION**

CONCLUSION

With the use of locally generated sequences, the analysis of different subsequence matches was explored using a popular technique in machine learning and neural network architecture

FUTURE RESEARCH

Instead of simply storing a logical TRUE or FALSE as the value corresponding to each k-mer, store the physical percent difference in the two alignment scores.

**APPLICATION**

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.

RESTRICTION SYNTHESIS:

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 match 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. Because of the algorithm’s high reliance on subsequence matching, this will be the main focus of the paper.

**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.