

A Deep Learning Model for DNA Sequence Prediction of Prokaryotes using an Artificial Neural Network

Kenneth D. Lemente, Glynn T. Pupa, Aprille A. Tiedra, Randy F. Ardeña
University of Mindanao, University of Mindanao, University of Mindanao, University of Mindanao

floyd_lemente@umindanao.edu.ph, heavenzero05@gmail.com,
aprilleshane_tiedra@umindanao.edu.ph, randy_ardena@umindanao.edu.ph

DOI: 10.26821/IJSHRE.9.2.2021.9214

ABSTRACT

Artificial Neural Network is broadly utilized in both academia and industry. It is a deep learning model with a capability to perceive a data's sequential characteristics and utilize patterns to predict the next likely scenario. Over the last few years, many types of research show how Neural Networks can be used for computational biology applications such as DNA sequence classification. A difficult problem that continuously remains in the wide field of biological community, is the correct gene prediction. A powerful predictive model can be a step towards developing more reliable gene prediction methods for DNA which would give a great advantage for the scientific community and researchers. Datasets released by the NCBI-NIH Gene Bank and Encyclopedia of DNA Elements (ENCODE) are acquired and used in this study as input data. The researchers used the encoding method of one hot vector in representing the sequences as input data to the algorithm where the non-coding variant DNA sequence will be encoded in a vector of integer, to transform our categorical labels into a vector of zeros and ones. The length of these vectors is equal to the numbers of the category that our model is expected to classify then the training follows. During the training phase, the embedded numbers will be mapped into substrings, also known as the regulatory motif, and are concatenated after.

Keywords: Deep Learning, DNA Sequence Classification, Prokaryotes, Regulatory Motifs, Machine Learning, Artificial Neural Network

1. INTRODUCTION

1.1 Background of the Study

DNA, a sequence of four letters (A, C, G, T), is an essential component in understanding life forms as it holds the instructions to our genes for the growth and function of humans, animals, bacteria, or simply, all of the organisms. Ever since the discovery of DNA, human beings have been aiming to understand the genomes of every species and decipher the biological data given by this genetic information through genomic research. Unlike genetics which studies on the specificity of genes, genomics focuses on the entirety of genes of an organism. Through the collaborative efforts made by genome projects such as Human Genome Project (2001) and ENCODE (2012), abundant data of DNA information are now available globally thus helping genomic research flourish. Prokaryotes, which include both bacteria and archaea, are found almost everywhere whether in our ecosystem, on the surfaces of our furniture, and within our bodies! Some are even found in conditions too extreme for other organisms, for example, hot vents on the seafloor [1]. Years of genomic research made it sure that the variety of bacteria is much broader than anticipated as shown by GenBank which currently holds 156686 genomes of prokaryotes, showing its diversity [2]. The advancement and increasing data intensity of genomics have prompted the progress of sequencing technologies, which makes it much easier to read DNA sequences.

As of late, Deep Learning has already accomplished significant performance in the field of information technology most especially in computational biology [3] with its known feature of effectively identifying complex

patterns from large datasets to train the neural networks that model high-level

Different deep learning methods were widely adapted to address fundamental genomics problems such as predicting sequence specificities and gene expression inference [4] and found that they showed favorable results than the traditional methods. In 2015, Alipanahi and Delong studied the sequences of DNA and RNA binding protein specificities through a model of deep learning [5]. In their research, they have a DNA sequence with varying lengths from 14-101 and they have achieved an average of 0.714 using five sequences with a diverse background. Another recent research of a model based on deep neural networks that discovers the DNA motif [6], used a three-stage approach to compare their study with previous investigations and achieved significantly better performance than the earlier researches. In these researches, one thing to notice is that they use multiple-layer network architecture to help combine the base instances into sequence motifs and also combine motif samples into more complex signatures trailed by connected layers producing a successful model of sequence specificity of protein binding that helped them learn the descriptive sequence of the signatures [7]. However, carefully chosen expertized features were used in these researches in representing DNA sequences that's why limited informational material from raw data was produced which led to the reduction of the architecture's execution. Because of this, predicting function directly from a sequence instead of selected datasets has been a growing interest in genomic research attributed to the fact that 93% of the approx. 6500 identified diseases and SNPs are found in regions that are non-coded [8]. Therefore, a flexible model predicting the transcription factor (TF) bindings accurately from DNA sequence with sensitivity of a single nucleotide and non - coding variant effects estimate may reveal a new understanding of these elements.

To address this fundamental problem, a hybrid model called DanQ was introduced in 2016 by researchers combining CNN and BLSTMs, a known form derived from Recurrent Neural Network or RNN [9]. RNN is another variation of DNN that have allowed researchers to address a range of issues in the domain of natural language processing. In contrast to CNN, a directed cycle is formed from the connections between units of an RNN which creates the network's internal state that enables it to display either temporal or spatial behavior. DanQ combines outputs of two RNNs that contain long - short term memory blocks instead of regular hidden units.

Having a simple approach like one convolutional and a maximum layer of pooling followed by the LSTM layer, it had shown better results than that of its previous researches.

In this study, the researches have created a deep learning model in predicting DNA sequences using the Artificial Neural Network (ANN) while seeing those sequences as data of text. The ANN will be a feed-forward neural network with hidden layers. These hidden layers are expected to learn the semantic relationship of each DNA sequences in response to its genome. With the help of a robust regression analysis algorithm such as Logistic Regression, the prediction will be much easier. Through this approach, the researchers intend to acquire greater accuracy in predicting DNA sequences of non-coding variants of Prokaryotes. This research aims to yield a novel understanding of genomes that are of non-coding sections that can be beneficial in the genomic research community especially in discovering complex variants. This thesis will be a comparative study to the GeneMark gene prediction program [10] which uses Markov Chain Model for predicting the DNA sequence of a genome and validate if using the approach would have a better performance than it.

We ask that authors follow some simple guidelines. In essence, we ask you to make your paper look exactly like this document. The easiest way to do this is simply to download the template, and replace the content with your own material.

1.2 Purpose and Description

This research main purpose is aimed at studying a non-coding variant DNA sequence of a prokaryote and its pattern to know some of its function and develop a deep learning model using a Feed-Forward ANN to predict the DNA sequence of a prokaryote. Through DNA sequencing, experts in metagenomics will be able to learn of the entire prokaryotic species with regards to their natural environment and function including those that cannot be cultured yet, unknown, and invisible to researchers.

The study was developed using a javascript framework known as Vue.js. The research is a comparative study against the GeneMark gene prediction program.

This research will determine whether an Artificial Neural Network model for predicting DNA sequences of a prokaryote will have better results than the GeneMark gene prediction program that uses Markov Chain Model to predict the DNA sequence of a prokaryote. Along with one hot vector encoding, the process of learning will be

much more natural for the deep learning model and with the unique characteristic of a feed-forward ANNs.

1.3 Objectives

The researchers established the accompanying goals and objectives for the study:

1.3.1 General Objective

The prime goal of this thesis was towards developing a deep learning model that predicts the DNA sequence of a non-coding variant of a prokaryote using an Artificial Neural Network.

1.3.2 Specific Objectives

To key target of this study was attained using the subsequent required specific purposes:

- 1.3.2.1 To study the possibilities of DNA sequence prediction through Supervised Learning.
- 1.3.2.2 To employ a Feed-forward Neural Network in generating the Genome information towards DNA sequence prediction.
- 1.3.2.3 To integrate a supervised learning strategy for training the neural network to exploit the DNA sequence of a prokaryote.
- 1.3.2.4 To prove that an ANN is capable of DNA sequence prediction with the help of One-hot encoding and Logistic Regression.
- 1.3.2.5 To validate the performance of the model in solving DNA sequence in comparison to the GeneMark gene prediction program using a simulation in Vue.js.

1.4 Scope and Limitations

The research will focus on the deep learning model training which will learn to predict the DNA sequence of a prokaryote. The model will use an Artificial Neural Network. The researchers will use public datasets that were used by other studies for training and will use a data set that will best suit the testing and validation. One limitation of this research is that the DNA sequence of a prokaryote can only be predicted if it has no mutated parts.

This limitation will affect the speed and process of training the deep learning model and the accuracy of the prediction.

2. REVIEW OF RELATED LITERATURE

During the beginning days, genes were found, developed and built with experimental organism authentication. At present, there are many computational methods widely used for DNA sequence prediction that have shown success in understanding, and modeling genome content

like identifying the functions and fixed positions (locus) of the genes, such as the Hidden Markov Model, eg. Van Baren et al. [12] that models the dependencies amongst the inline bases; Support Vector Machine method presented by Kim et al. [2] in recognizing subtle relationships among expression profiles to predict their target transcript factor; and Machine Learning, Hoff et al. [14], that used grammars which are context free and extensive training to provide a gene prediction algorithm based on machine learning with a method of two stages for metagenomic materials.

A Deep Learning Model using an Artificial Neural Network was presented by the researches to provide a tool for creating an algorithm that predicts the DNA sequence of a non-coding variant of a prokaryote. The following are related literature for the study:

2.1 GeneMark: Parallel Gene Recognition for Both Strands

It is widely known that models of Markov Chain calculate the likelihood that a specific nucleotide will appear after a sequence to obtain its statistical information. This study combines the specific Markov models with Bayes' formalism to assess whether the fragment in question is a portion of region either non-coding or coding. The idea is to incorporate a Bayesian algorithm to avoid the common drawback of methods under gene finding that generate artificial signals when the actual coding region is found on the strand of DNA complementary to it. This Bayesian function uses a Markov model (non-homogeneous) of the sequence complementary to the protein-coding sequence also referred to as the coding region's shadow [17]. The Markov chain model is firstly used in this study with relevance to finding sequences of DNA material, and its success resulted in furthering research in this direction.

The above literature will use a deep learning model in predicting the DNA sequence's non-coding variant. In contrast to the study above, the research will use Logistic Regression to calculate the probability of the predicted DNA sequence and Softmax Function will be applied if needed.

2.2 Discovering the Optimal Segmentation of the Secondary Structure of an RNA with the use of Dynamic Programming

Dynamic programming is an optimization technique that can be used to solve problems in optimization by combining solutions to sub-problems. In this study, the algorithm was used to discover the best way in

segmenting long RNA sequence optimally into non-overlapping chunks to maximize the similarity of the observed structure versus the predicted structure. This approach is great for overcoming storage constraints [18].

In the project, the Feedforward ANN will be trained for the optimization of predicting the DNA sequence(output). In contrast to the study above, the research will use Vector Representations of text such as one hot vector encoding for the segmentation of the DNA sequence.

2.3 B.L.A.S.T. (Basic Local Alignment Search Tool)

B.L.A.S.T. is a group of programs which uses a basic search means of local alignment that was intended for rapid sequence comparison that uses the subset of a sequence and attempts to align it to the subset of other sequences in order to discover regions of biological sequences with similar properties [19]. It calculates statistical significance of found matches by comparing the given sequence to sequence databases [20].

As opposed to BLAST, the study will use new DNA sequences as the basis for the output using a Neural Network. The research will predict the DNA sequence instead of using subsets of sequences and finding similarities in a sequence to produce an output.

2.4 Deep Learning Method for DNA-RNA Prediction

It is important and critical to know the particular arrangement or specifics of DNA-RNA proteins in order to develop representations of the regulated or monitored methods in the vast structures of biology, which can also be of use in identifying variants of casual diseases. In this study, they used CNN in an approach called DeepBind to show that these sequence specificities can be found out using the deep learning technique in the form of a readily visualized mutation map. In here, they used a various array of experimental data to provide a computational method in discovering patterns [21].

Similar to the study above, this research uses deep learning for prediction. In contrast to the study above, the research uses Feedforward ANN as its Neural Network, Logistic Regression for prediction analysis, and One hot vector encoding as its text classification and representation.

2.5 A Deep Learning Model: Discovering the Effects of Non-coding Sequences

DeepSEA is a program that uses deep learning architecture to accurately predict Transcription Factor sites of proteins and histone marks using only the given DNA sequence as input to the model. Its ability to predict with high accuracy helped guide further researches in evaluating the functional significance of non-coding variants [22].

The study and the study above both use deep learning model, a non-coding variant DNA as its input, and logistic/sigmoid function to predict output and its probability. In contrast to the study above, the research uses Multilayer Feedforward ANN while the study above uses Multilayer Convolutional Neural Network.

2.6 A Hybrid of CNN-RNN called DanQ

DanQ utilizes a hybrid architecture of neural networks which combines CNN-RNN, for discovering the function of a given strand that is non-coded. In this study, the convolution step is using convolutional filtering to locate motif sites in the scanned sequences, and a bi-directional long short-term memory was used subsequently for the consideration of spatial orientations between motifs [25]. The method showed how DanQ enabled simultaneous learning of motifs and the regulatory grammar between them thus making it a robust process in understanding non-coding DNA [23].

The study above uses a Multilayer Hybrid Neural Network with LSTM as its variant to predict the DNA sequence. In contrast to the study above, the model will use a Feedforward

3. TECHNICAL BACKGROUND

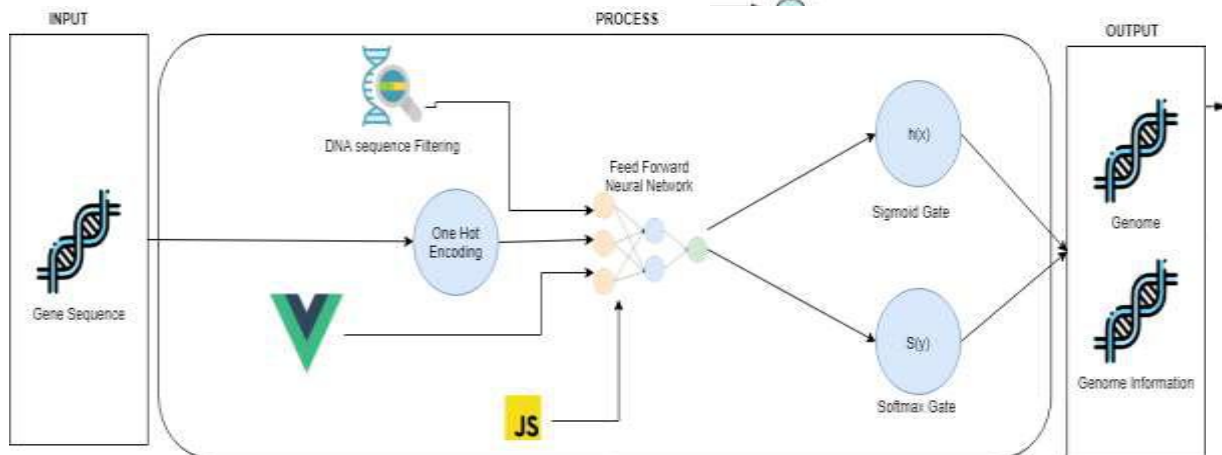
The study is to create a model that uses deep learning algorithm that will learn and predict the DNA sequences of the non-coding variant of a prokaryote. The researchers will use an Artificial Neural Network. The input will be DNA sequences in forms of text (A, T, G, c) encoded into numbers using the encoding method called One-hot vector. The numeric values are then passed on to the Gated Recurrent Unit which are concatenated starting from the smallest embedding size. Then the ANN will filter the possible DNA sequences and will be used for the Logistic Regression Block in analyzing the best possible DNA sequence. Then the Softmax Function will output the predicted DNA sequence.

3.1 Conceptual Framework

The study's main goal was to create a model that uses a deep learning model algorithm which is the ANN in order to search and find the DNA sequence of the non-coding variant of a prokaryote, is implemented through the use of JavaScript and Vue.js framework.

The researchers have measured that the deep learning model used has been ideal and essential in achieving the primary purpose of this study.

Figure 1 Conceptual Framework of the Study



The input of the system is a substring of the DNA sequence of a prokaryote in forms of text and will be converted to a binary vector using one hot encoding before going through the ANN block (training and testing) and K-mers mapping(testing). The binary vector will then be filtered and will go to the Logistic Regression Block for prediction analysis of the DNA sequence before producing an output.

3.2 Artificial Neural Network

The idea behind ANNs is to map and connect the neurons with human brain's neural structure as the basis for the linkage. ANN records one-by-one and learns by looking at its anticipated output with the actual record in which the initial error prediction is returned to the network and is used in the next iteration to modify the network algorithm [28]. The process will then be repeated multiple times.

Artificial Neural Networks are very powerful and showed success in different fields of application such as pattern recognition, statistical mapping, modeling, and prediction. A necessary ANN is also known as Feedforward ANN because it does not involve any backpropagation. The ANN's performance would depend on its input, hidden layer, and its output.

The figure overhead shows ANN's process of the hidden, input, and output layers together as a whole. The input neurons represent the information of data as an input of which the researchers are trying to predict or classify. The hidden layer where the weighted inputs are taken and will produce an output through an enacted work: activated function. Layer of the anticipated yield was where network coalesces or produces the result.

3.3 Application of Neural Networks on Genomics

One of the challenges in the field of genomics in today's time of big data was the critical need for the ability to transform this big data into valuable knowledge. Since the early 2000s, Deep Learning has emerged rapidly and accomplished significant advances in numerous fields such as NLP and image-speech recognition. It has become well-known in the biomedical community due to its up-to-date capability in extraction of progressive depictions hierarchically and identification of intricate patterns from large datasets to train the neural networks. Different deep learning algorithms have their functions to solve different types of problems in genomics [24]. For instance, convolutional neural network (CNN) famous for its capturing ability in classification tasks was used in a study to predict the specifics of DNA and RNA-binding proteins [26]. To understand effects of non-coding variants for prediction of chromatin features

[22], a study in 2015 used deep learning model to accept genetic sequences as data inputs [27]. As by Quang and Xie (2016), a hybrid of a CNN-RNN method was used to quantify the function of non-coding DNA in a program called DanQ [23]. As deep learning continuously showed successes, new deep architectures such as hybrid models are employed to this day in approaching genomic problems.

Figure 3 From top to bottom: Sample outputs of applications in genomics that uses Neural Networks (DanQ & DeepBind)

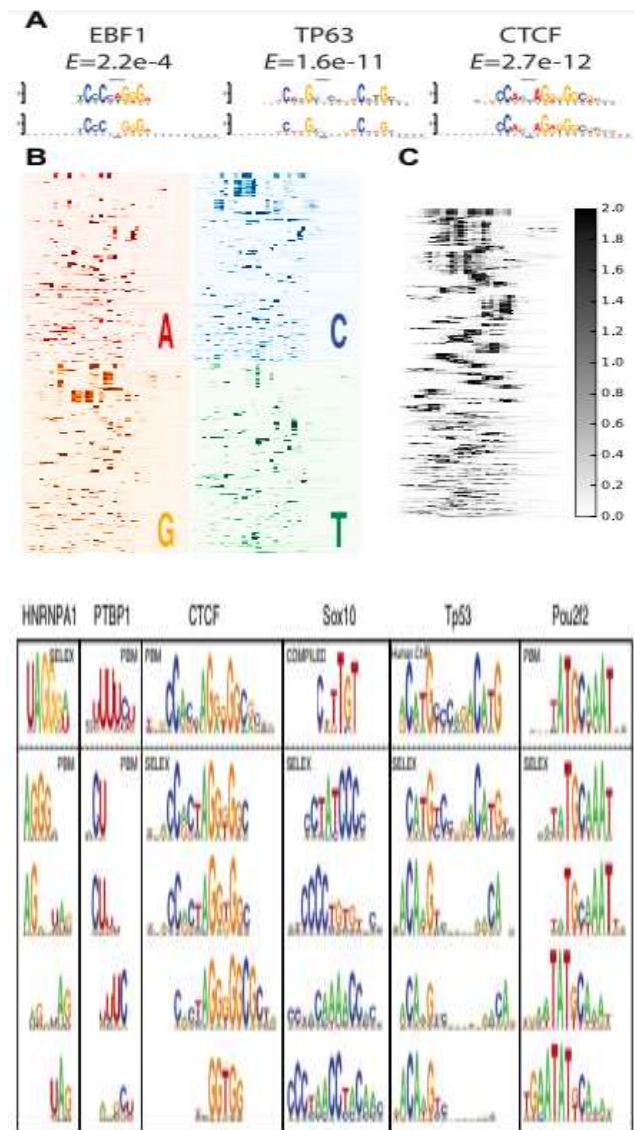


Figure 3 shows the result of different researches. The application was implemented and experimented using Neural Networks. It shows above that Neural Networks are very useful to the community of genomics, specifically in the robust analysis of DNA sequences.

In the research, the approach would be the same as the above figure, i.e. training an AI model in finding the best solution and demonstrate the positive effects of AI in genomics.

3.4 Workflow Model

Figure 4 Workflow Model

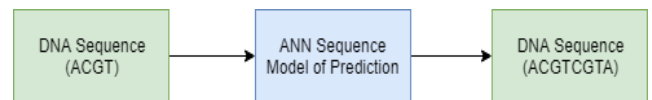
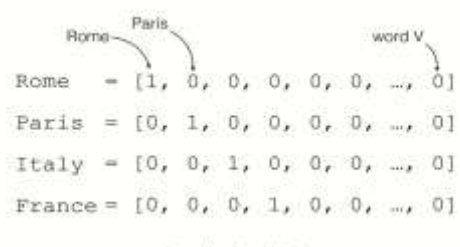


Figure 4 shows the prediction model in which the input is a 4-length DNA sequence and the input is included as each initial sequence where the output is in a length of 8. The sequence does not necessarily have to be twice the length of the input; instead, it varies accordingly on the learning capability of the ANN, the gathered data, and the functionality of the input.

3.5 Word Embeddings for Text

Word embedding is a kind of representation of words that allows similarly meaningful words to be represented in the same manner. Word embedding is characterized by the distributed text representation that may be one of the fundamental discoveries that recognizes the notable performance of deep learning algorithms in handling the challenge in natural language processing (NLP) field.

Figure 5 Sample One hot Vector Encoding

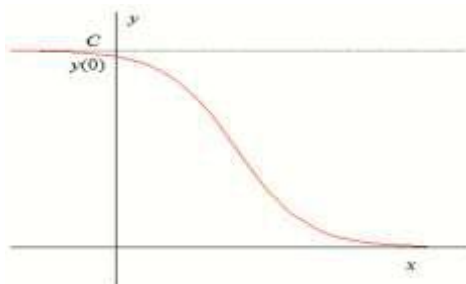


To put the DNA Sequence into the machine learning algorithm, the text data are converted into a vector representation. In this stage of the system, all the text in a DNA sequence will be encoded as a binary vector using one hot vector encoding. One hot vector encoding allows the data to be more expressive. One hot vector encoding maps all possible inputs from char/word values to integer values. Let's start with an example with the word "Italy," the word Italy in the input is encoded as 3, or index 3 in the array of possible input values. The integer encoding is then encoded to one hot vector wherein a list of 0 values is

created with the length of the possible inputs so that any expected char/word can be represented. Next, all the indexes of each char/word are marked 1.

3.6 Regression Analysis

Figure 6 Graph of Logistic Regression Function



Logistic regression also known as the binomial regression, is a statistical approach in dataset analysis in which a result is determined by independent variables. This predictive analysis is suitable if the dependent variable is dichotomous hence, binary. It is used in describing data and explaining the relationship between a binary dependent variable and independent variables that can be ordinal, ratio, nominal, or interval [29].

$$y = \frac{C}{1 + Ae^{-Bx}}$$

As observed above, the sigmoid function (logistic) contains parameters namely (A,B,C). The input x will grow, if all the parameters are positive constants, and the exponent -Bx becomes a more significant negative value. The denominator, therefore, always exceeds 1 and decreases to 1 as x grows large meaning the value of y is still lower than C and increasing to C.

Logistic functions are appropriate methods in observing growth in the population of species that have grown to such an extent in saturating their ecosystems.

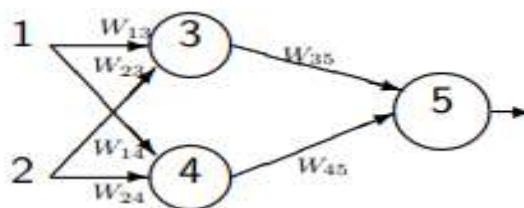
$$S(y_i) = \frac{e^{y_i}}{\sum e^{y_j}}$$

The formula above shows the general form of the Softmax function which is a generalization of the sigmoid function. It squashes all the arbitrary real values of a K-dimensional vector to logits and turns

them into probabilities that sum to one. It is used as an activation function if the output of the final layer cannot be interpreted as the sigmoid function or undergoes log loss. The function can also be used in classification.

3.7 Network Topology

Figure 7 FFNN Topology Example with Given Weights



$w_{13} = 2$	$w_{35} = 2$
$w_{23} = -3$	$w_{45} = -1$
$w_{14} = 1$	
$w_{24} = 4$	

A set of weights is defined for every node in the neural network by w_{in} w_{in} . In the figure above, node 4 has a set of weights which are w_{14} and w_{24} . By determining the set of all weights and its activation function or its topology we can define the network.

Based on the figure above, what would be the output of the network if we used two inputs $x_1=1$ and $x_2=0$. To calculate the final node's output, we must first calculate the weighted sum of the first hidden layer utilizing the formula $v_n = w_{in} x_1 + w_{in} x_2$.

Therefore,

$$v_3 = w_{13}x_1 + w_{23}x_2 = 2 \cdot 1 - 3 \cdot 0 = 2$$

$$v_4 = w_{14}x_1 + w_{24}x_2 = 1 \cdot 1 + 4 \cdot 0 = 1$$

Next is we apply an either of the activation function which is sigmoid or softmax. But in the case above we will use the sigmoid function:

$$f(v) = \frac{1}{1 + e^{-v}}$$

The output of the function is $y_3 = f(2) = 1$, $y_4 = f(1) = 1$. Then we get $v_5 = w_{35}y_3 + w_{45}y_4 = 2 \cdot 1 - 1 \cdot 1 = 1$ which is the weighted sum of the last node. Then we apply the

activation function to the previous node to get the output:

$$y_5 = f(1) = 1$$

The output will be used as data for the deep learning model as one of its training set to compute the errors in the output and learn to get the desired output.

Figure 8 Encoder-to-Decoder Sequence-to-Sequence

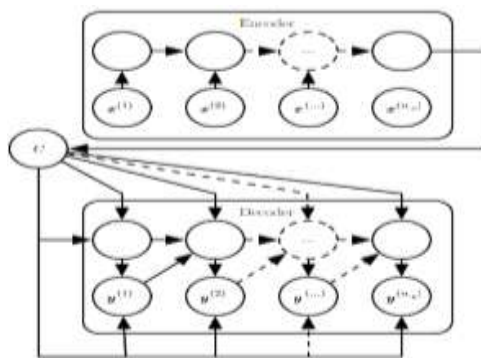


Figure 8 is an example of an encoder-decoder for learning to generate an output sequence ($y_1 \dots y_n$) from the input ($x_1 \dots x_n$). The figure above includes an ANN known as RNN that could create the probability of a given output sequence or produce an output prediction wherein using the final hidden state of the RNN encoder, a generally fixed context variable C is calculated. It is a semantic summary of the input sequence and is given to the RNN decoder as input [30].

3.8 Prototype

Figure 9 From left to right: An example output of the system prototype.

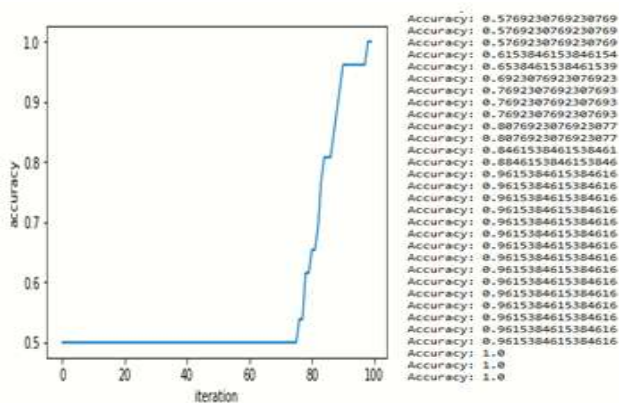


Figure 9 shows the output of the prototype in which it shows in the graph that there is a fluctuation in the

accuracy of the prediction model when more data is feed to it, but the result still had perfect accuracy. The researchers used 50 DNA inputs as the training set and two types of genomes as the initial dataset and basis for the accuracy test.

3.9 Development Tools

The researchers decided to build an Artificial Intelligence that can predict the DNA sequence of a prokaryote and provide its genome with a Neural Network with Feedforward topology. The researchers will use the following different tools to carry out this study:

3.9.1 Vue.js

Vue is a progressive framework for the construction of user interfaces. This framework is distinct to other monolithic frameworks because of its nature to be incrementally acceptable. Its core library focuses solely on the view layer and pick - up and integration with other libraries or existing projects is relatively easy. On the other hand, when used in combination with modern tools and supporting libraries, it is perfectly able to power sophisticated applications with single page [31].

3.9.2 JavaScript

JavaScript is a is a multiparadigm, prototype - based, dynamic language that supports object-oriented, imperative, and declarative styles [32].

4. METHODOLOGY

This chapter provides the methodological framework of the study. It presents and discusses detailed reports of the methodologies. Generally, it explains how the researchers used the algorithm in every module to achieve the goals and objectives of the study.

4.1 Data Gathering

In the development of the study, different concepts and ideas were taken account in coming up for the best method to develop the study of developing a model of deep learning to predict the DNA sequences of a non-coding variant of a prokaryote using an Artificial Neural Network. The researchers explored different Gene Banks and decided to go with the NCBI – NIH combined with the gathered resources and other web-based information in relation to the study. Different methods, algorithms, any possible procedures were also considered for the collection of data. Additionally, the researchers chose the programming language, applications, and materials to be utilized so that the study can be made. The researchers will likewise recommend by the end of the study.

Table 1. Sample Dataset

Genome	Gene/DNA Sequence in FASTA format
Smorhizobi Pseudomo um meliloti has 1021. aeruginosa PAO1	ATGCTGCTGCTTGGACCTGCTGTATA ATGCTAECGTAAACTCGCTGCGGTAT CGTTTGAACCTGGCACAAATACAAC CCCTTGGTGTCCCTGCTCACTGGCC TGTGGAAGGCATCGCCGGACGCTGGT ACTGCTGGCTGCGGAGTGCFCGGT TTGACCTGACCACGTTTGTATGATGA GACATCCAGGGTAACCGACCAATG GCGGACTTTTTCGGCCGCTCCGCT CAGTTCACACCAATGCCATCACCG GCTCTTCACCAGGATGAAAGCGGATC TCGACAAGAGCTGCAAGCAGTTC CCGTCAACCTGTCCCAACCCGGGCA GGGATTCCCGGGGAATATGGCCTCT CCTGCCGAAGAACTGCATGGGCECA CAACCTGGTACTGAGCACCCGCGCC TTGAAGGCTTCCGCCAGCCACCGGA GACATGCGAGGGCGTGGTACCCGAC TGAAGGCTGCGAAGAGGCTTATCGT GGCATGGCTTCCGGCTGGACAAG CTATTGGTGGAGGATACCGGTGAG GATTACCTGAGCCCGACGACAGC CCGGATTTTGGACACTTCCGGCATG CCTGCTGGGGGGAGGCTGACAGTG TCGCTTGGGCGAAGAGGACTGGG AGGAGGCTTTTGGGTTGAGTTCCG TGACCTTCGACCTTCACAGCTGAA CAGTGCATACCGGCTGCTGGGTGAC GGAAGGCGAGCAGTACATGTTCTC GTCCCGGAGACCGGTGCGGCTCTATT TGCACCTTCCCGGGCCACTCCGGCC TTGACTATGAGGCGTATGCGCGGGA TGATGAAGGGCACCCGACCCCTGA AGTCA TTGACGGTCATGTGTTCCGTCGGTG
	A

4.2 System Design

This area shows the framework design in preparing and developing of the study created by the researchers.

4.2.1 System Input, Process, Output (IPO)

In this area, researchers developed a conceptual structure (figure 1) which represents the IPO process of the model. The conceptual design would serve as assistance to the researchers define and understand its structure, flow, and outcome of the study. The input will consist of the Pre-training stage where all the necessary data such as the DNA Sequence in FASTA format with its symbol to train the model and the Prediction Stage where the user can dynamically input a DNA sequence of a Gene in FASTA. The data in the pre-training stage will go to several processes. The output of the project will be a probability of the most likely predicted Genome.

4.3 Development

As a method for developing the project, the researchers decided to use the Agile Software Development Methodology. The general idea of this approach is to develop specifications and solutions using cooperation of the self-organization and cross-functional groups

with inputs coming from their customers. The researchers developed the requirements for the study after thorough discussions and collection of data from the selected resources.

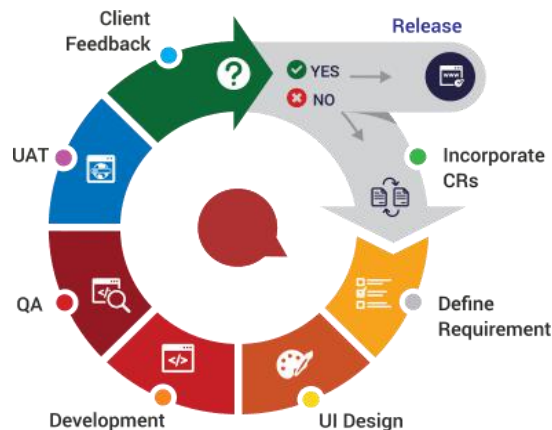
Figure 10 Methodology for AGILE Development

Stage 1: Analysis of System Requirements

From data gathered, all the relevant information was collected by the researchers about ANN and DNA sequence prediction, as well as the Gene banks used for training the model from the web and other resources that would be helpful for the study. The researchers also gave different suggestions and shared their knowledge of the research tools and design required. Afterward, they analyze and study the requirements and confirm them in a group discussion with the adviser so that the analysis can be perceived if it is legitimate.

Stage 2: Design Analysis

When all the information required for the research study were collected by the researchers, the Conceptual Design was made to analyze the flow of



the project process. After analysis of the design, the project's development became faster since the researchers are already aware of the requirements as well as the flow of the process. The developer of the group created the algorithm design and some minor designs of the UI. The rest of the team contributed to gathering the data needed and processed the training of the model to find the best-case scenario for the model. The developer used Vue.js for the design of the UI and applied the algorithm and FFNN with the help of a data-flow open source library.

Stage 3: System Advancement and Implementation

Afterwards, defining the requirements and analyzing the design were made. Subsequent phase was to use it in coding format. First, the researchers needed to load node.js for the installation of the vue.js and its node modules then install it. After installing node.js the developer runs a command to create a vue.js framework.

4.4 Modules and Procedures

This chapter explains how each system module uses the algorithm for the study made by the researchers.

4.4.1 Training Module

The training module describes the initial part of the development where the objective is to supply the model with gathered data. The input data sets for the training produces an output that is used by the accuracy module.

4.4.2 Accuracy Module

Accuracy module will show the prediction accuracy of the module in every iteration of data when the training is done. The data used in this module are the outputs of the training module and is mapped to the graphical output of the system.

4.4.3 Prediction Module

In this module, the user will input two (required for greater accuracy) DNA sequences in FASTA format to produce an output (Genome). The output of the system is a probability of the input to a Genome. The highest probability rate is the chosen genome as the predicted output.

4.5 Implementation

This chapter explains how each system module uses the algorithm for the study made by the researchers.

4.5.1 Implementation of the Feed-forward Neural Network Model

The goal of a Feed Forward Neural Networks is to approximate a function which is the error function. During training, the FFNN maps all the values of input x where all the information flows through and is being evaluated and is mapped to the output y. By mapping all the values of x to y, an error function is produced in each iteration of the training. The code below shows the mapping of inputs to an output, definition of the mode, layer's amount used in training, and the approximation of its error function.

Figure 11 Input

```
{ "id": 0, "symbol": "yjhr", "input": "ATGGGATATTTACATATTGATGGTAGAGGCATGAAT"
{ "id": 1, "symbol": "nfra", "input": "ATGAAGGAGAATAACCTTAATCGCGTCATCGGATG"
{ "id": 2, "symbol": "thr1", "input": "ATGAAACGCATTAGCACCACCATTACCACCACCATI"
{ "id": 3, "symbol": "insb-1", "input": "ATGCCGGGCCAAGCCCGCATTATGGCCCTTGG"
{ "id": 4, "symbol": "sspa", "input": "ATGGCTGTCTGCTGCCAACAAACGTTCCGTAATGAC"
{ "id": 5, "symbol": "yaaJ", "input": "ATGCCAGATTTTTTCTCCTTCATTAAACAGCGTCTCT
```

The code in Figure. 11 is a simple input in a JSON format and is given an id so that it can be used as a reference during the mapping of output.

Figure 12 Output

```
{ "id": 0, "genome": "escherichia coli"},
{ "id": 1, "genome": "pseudomonas_aeruginosa_pao1"},
{ "id": 2, "genome": "Caulobacter_vibrioides_CB15"},
{ "id": 3, "genome": "Vibrio_cholerae_01"},
{ "id": 4, "genome": "Sinorhizobium_meliloti_1021"}
```

The code above is a sample of output and will be used during the mapping.

Figure 13 Mapping of Output to Input

```
{ "x1": 749, "x2": 750, "y": "Caulobacter_vibrioides_CB15"},
{ "x1": 750, "x2": 751, "y": "Caulobacter_vibrioides_CB15"},
{ "x1": 801, "x2": 802, "y": "Vibrio_cholerae_01"},
{ "x1": 802, "x2": 803, "y": "Vibrio_cholerae_01"},
{ "x1": 803, "x2": 804, "y": "Vibrio_cholerae_01"},
```

The figure above is a sample mapping of x input referenced through the use of id as per stated in the previous figures to the output y, a genome.

4.5.2 Application of the Logistic Regression

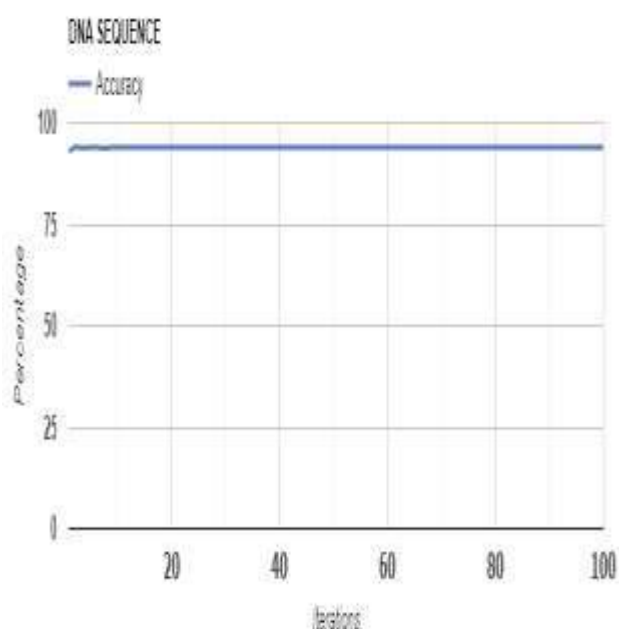
After receiving the Feed Forward Neural Network output which is the minimum error function in weights,

Logistic Regression is then applied. The logistic function or sigmoid function will be the activation function of the hidden layer during the training of the AI. The logistic function squashes the calculated error function so that at each iteration of the training there is a clear difference in the error function. Which then the error function will be used as a basis for the prediction of the probability. The code below shows the backpropagation and the number of iterations during training and the logistic function algorithm.

4.5.3 Graphical Output

The algorithm was implemented using a JavaScript framework Vue.js. The training sets for the algorithm were captured from the NCBI (National Center for Biotechnology Information) genetic bank. The training sets consist of 5 different genomes and 1000 different protein-coding genes. The size of the current training set is limited. Thus, creating a higher model for prediction is impossible. The graph shown after the training is known as the accuracy rate of the prediction during each iteration. The accuracy rate is the approximation of the error function and the activation function. The values shown in the graph below are taken directly during the training process of the AI, but it does not define the efficiency of the algorithm.

Figure 14 Accuracy Rate Graph



4.5.4 Accuracy of the procedure

The figure above (see Fig. 14) is just an example of the graphical output of a method defined in the system. The evaluation of the quality of the prediction model is done through the elimination of factors that can create the worst-case and best-case scenario for the prediction model. During the best-case scenario, the model is trained to 3 types of a genome and a data set ranging from 15, 50, 100, 150, and 200 types of protein-coding DNA per genome. The accuracy rate shown below is an average of the final accuracy rate after training the model thrice with the same number of data sets.

Table 2. Accuracy Rate

No. of Data per Genome	No. of Genome	Accuracy Rate
15	3	98%
50	3	95.18%
100	3	95.32%
150	3	95.15%
200	3	95.12%

4.6 Testing

The researchers carried out a series of tests to measure the system's efficiency and effectiveness. The test series is as follows: Performance Testing and Software Testing.

4.6.1 Software Testing

The researchers conducted this testing to test the utility of the software using the Feed-forward Neural Network and Logistic Regression. This determines whether the system is fully operational and ready for deployment.

Table 3. Software Testing

PROCESS
Features and functionality of Logistic Regression in the model can be tested?
The Feed Forward Neural Network is functioning?
Both algorithms can be used to create a prediction probability?
The Function for prediction will process the Input if it is valid
The Function for prediction will not process the Input if it is invalid

Figure 15 Training Progress



The figure above shows the training progress of both the old and new algorithm during the simulation of the system.

4.6.2 Performance Testing

The researchers test the run-time performance of the software and measured its stability and responsiveness under a certain load of both the old and new algorithm using different Hardware specifications.

Table 4. Hardware Testing

Component	PC 1	PC 2
Model	MSI GL 62	Lenovo ThinkPad
CPU	Intel Core i5(3.2GHz) 6 th Gen	Intel Core i5 (2.6GHz) 3 rd Gen
RAM	8gb Ram	8gb Ram
Storage	500GB SSD	200GB SSD

5. RESULTS AND DISCUSSIONS

The researchers will deliberate and talk about each test’s results during the development of this chapter.

5.1 Results of Software Testing

Table 5. Software Testing Results

SOFTWARE TESTING RESULTS		
PROCESS	YES	NO
Features and functionality of Logistic Regression in the model can be tested?	✓	
The Feed Forward Neural Network is functioning?	✓	

Both algorithms can be used to create a prediction probability?	✓	
The Function for prediction will process the Input if it is valid	✓	
The Function for prediction will not process the Input if it is invalid	✓	

The figure above shows the result of software testing. The result shows that the system is fully functional and is working. Thus, the system is ready to be deployed, and its performance can be tested.

5.2 Performance Testing Results

The researchers tested the new and old algorithm on different hardware specifications to see if there is any difference in the performance of both algorithms.

Figure 16 Performance Result of the Old Algorithm



The figure above shows the result of the old algorithm’s performance test after training it with five different Genomes and each Genome had a total of 200 different Gene sequence as our training data set. The graph s the rise and fall of the accuracy rate of the old algorithm which is HMM.

Table 6. Performance Test Results of Old Algorithm with 5 Genomes using Lenovo ThinkPad

No. of Data Per Genome	No. of Genome	Avg. Training Time in S	Accuracy
15	5	9.8s	84.67%
50	5	46.21s	73.96%
100	5	81.82s	73.97%
150	5	124.41s	82.95%
200	5	170s	83.79%

Table 7. Performance Test Results of Old Algorithm with 5 Genomes using MSI GL 62

No. of Data Per Genome	No. of Genome	Avg. Training Time in S	Accuracy
15	5	8.8s	84.67%
50	5	42.13s	73.96%
100	5	80.2s	73.97%
150	5	109.41s	82.95%
200	5	142.7s	83.79%

The tables 6 and 7 shows the result of the old algorithm when trained in the Lenovo ThinkPad and MSI GL 62. The training was conducted with different number of genes per Genome. The result shows that the accuracy rate is inconsistent because the accuracy rate itself fell when the number of genes per Genome was increased from 15 to 50 and 50 to 100. The accuracy rate then went up after increasing the number of genes per genome from 100 to 150. The graph from figure 16 likewise demonstrates the accuracy rate of the old algorithm was inconsistent.

Figure 17 Performance Result of the New algorithm



The figure above shows the result of the new algorithm's performance test after training it with five different Genomes and each Genome had a total of 200 different Gene sequence as our training data set. The graph shows a consistent accuracy rate.

Table 8. Performance Test Results of the New Algorithm with 3 Genomes using MSI GL 62

No. of Data Per Genome	No. of Genome	Avg. Training Time in S	Accuracy
15	3	6.2 s	98%
50	3	11.7s	95.18%
100	3	22.33s	95.3%
150	3	31.7s	95.15%
200	3	40.2s	95.12%

Table 9. Performance Test Results of the New Algorithm with 3 Genomes using Lenovo ThinkPad

No. of Data Per Genome	No. of Genome	Avg. Training Time in S	Accuracy
15	3	7.4s	98%
50	3	14.72s	95.18%

100	3	25.33s	95.3%
150	3	38s	95.15%
200	3	47.96	95.12%

The results in tables 8 and 9 are trained with three different Genomes with a different number of data per Genome. The results show that the accuracy rate of the new algorithm is consistent and high. The accuracy rate is consistent because the difference in each training is close to each other.

Table 10. Performance Test Results of the New Algorithm with 5 Genomes using MSI GL 62

No. of Data Per Genome	No. of Genome	Avg. Training Time in S	Accuracy
15	5	9.21s	85.35%
50	5	18.79s	83.96%
100	5	40.12s	83.97%
150	5	53.71s	93.96%
200	5	69.92s	93.94%

Table 11. Performance Test Results of the New Algorithm with 5 Genomes using Lenovo ThinkPad

No. of Data Per Genome	No. of Genome	Avg. Training Time in S	Accuracy
15	5	11.01s	85.35%
50	5	22.21s	83.96%
100	5	41.89s	83.97%
150	5	60.01s	93.96%
200	5	81.04s	93.94%

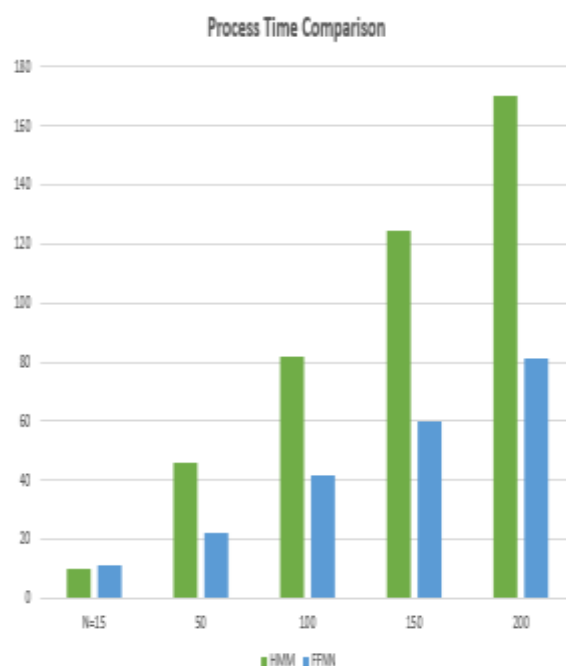
The results in tables 10 and 11 are trained with five different Genomes and are trained with the same

environment in tables 6 and 7. The results show the average training time and the accuracy rate of the new algorithm in a certain workload. The accuracy rate is also consistent.

The results of both PC models depict that with the use of a PC model with higher hardware specification the lesser the average training time of the new algorithm compared to the old algorithm. The accuracy rate of the new algorithm is much more consistent and higher than the old algorithm.

5.3 Process Time Comparison of the two Algorithms

Figure 18 Process Time Comparison Graph (HMM and ANN)

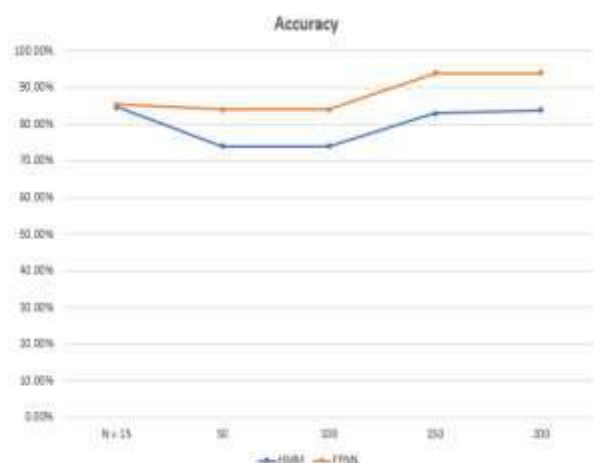


The graph in figure 18 is a comparison of the two algorithms concerning the processing time or training time of each algorithm. The y-axis of the graph is the training time in seconds, the x-axis is the number of inputs per Genome, and there are 5 Genome. The results show that during the training time for the first time with 15 inputs per training the old algorithm has lesser the time but as the quantity of inputs rose, the training period of the process would be lesser. As shown in the final training, the HMM algorithm's training time would be more than twice that of the

algorithm showing that the old algorithm has a better performance to training time than the old algorithm.

5.4 Accuracy Rate Comparison of the two Algorithms

Figure 19 Accuracy Rate Comparison Trend (HMM and ANN)



The chart in figure 19 shows the comparison between the accuracy result of the two algorithms after the training. The y-axis is the accuracy rate in percentage, the x-axis is the number of inputs, and a total of 5 genomes per accuracy rate. The comparison clearly shows how much a big difference there is the accuracy rate of both algorithms. The accuracy rate of the algorithm did not drop to less than 92% even with the increasing number of inputs per Genome.

5.5 User Acceptance

The researchers carried out a User Acceptance Evaluation to determine the functionality and usefulness of the system by the user's perspective. The criteria for the evaluation are the Functionality, Reliability, Correctness, and Responsiveness of the system software. A total of ten different users with a background in software development evaluated the system where each can give a score from 1, as the lowest, up to 5, the highest. Table 6 shows the evaluation result.

Table 12. User Acceptance Evaluation Result

Respondent	Functionality	Reliability	Correctness	Responsiveness	Appearance	User-Friendly
1	5	4	4	4	4	4
2	5	4	5	4	3	5
3	5	5	4	5	4	4
4	3	5	4	5	2	4
5	5	3	5	4	3	3
6	5	4	4	4	4	4
7	5	3	4	5	4	4
8	4	5	2	4	4	5
9	5	4	4	4	3	5
10	5	4	4	4	4	5
Ave	4.7	4.1	4	4.3	3.5	4.3

The overall rating for the system is shown in te table 6; The results show that the appearance of the system has an overall poor rating. Functionality wise, the system proved it to the users that the results of the prediction are correct, and the accuracy rate also shows how reliable the prediction is.

5.6 Summary of Results

Different types of testing were made to achieve the results as stated in the different figures above. The software testing showed that the functionality of the logistic regression in the model is working. The testing also showed that the FFNN is functioning. Next, it showed that both algorithms could create a prediction probability. Additionally, the system will only create a prediction when the input is valid.

In the performance testing, the researchers tested both the new and old algorithm on different PC Models (MSI GL 62 and Lenovo ThinkPad) with different hardware specifications. The new algorithm was tested on two different scenarios. First, the researchers trained the new algorithm with three different types of Genomes and a range of data sets (15, 50, 100, 150, 200). Second, the researchers increased the number of Genomes to 5 with the same range of data sets. The old algorithm was trained as well with five different Genomes and the same range of data sets. The researchers compared the result of the second scenario of the new algorithm and the results of the old algorithm. The comparison showed that the new algorithm has lesser average training time and a higher accuracy rate than the old algorithm.

6. CONCLUSION AND RECOMMENDATION

After testing the performance of the new algorithm. The researchers reached a conclusion.

6.1 CONCLUSION

The software testing for the functionality of the model went smoothly, and the performance testing results were above the expected accuracy rate and had no problems. It is also observed that there is a sudden drop in the accuracy rate of the old algorithm when training it with a range of 15-50 inputs. The comparison between the old and new algorithm shows a big difference in the processing time, and the accuracy rate is more consistent in the new algorithm. Therefore, through supervised learning, DNA sequence prediction is possible and employing Feed Forward Neural Network in generating genome information can lead to better DNA sequence prediction. Additionally, the researchers conclude that the performance of the new algorithm is better than the old algorithm, but there were limitations in the number of genomes the researchers could use as a data set because the more genomes there is, the lesser the accuracy rate.

6.2 RECOMMENDATIONS

As the comparative study showed promising results using supervised learning, the researchers believe that DNA sequencing prediction can be further improved with it. The limitations of the new algorithm can hinder the improvements of the prediction algorithm as concluded by the researchers and the dataset used was deemed fit for both algorithms. The researchers recommend using supervised learning for prediction analysis. Additionally, the researchers suggest that future reference of the research may include the use of Recurrent Neural Networks. Also, it is essential to investigate the sudden drop in the accuracy rate of the old algorithm (see Fig. 17) as to what caused it and what appropriate method should be used to prevent it for future reference. The researchers also recommend on using dynamic learning strategy for future development of the system, which adapts to changing data during training and allow file inputs because some of the DNA sequences are very long. Lastly, the researchers suggest venturing alternatives for predictive analysis like Logistic Regression and using other methods to improve the algorithm's efficiency and its limitations in comparison to other well-known algorithms in DNA sequence prediction.

7. ACKNOWLEDGMENTS

The researchers would like to express their deepest gratitude and appreciation to the parents, subject teacher, adviser and panel of experts for their unreserved support, guidance and patience which made this study successful.

8. REFERENCES

- [1] Prokaryote classification and diversity. (2012). Retrieved 2018, from <https://www.khanacademy.org/science/biology/bacteria-archaea/prokaryote-metabolism-ecology/a/prokaryote-classification-and-diversity>
- [2] Clark, K., Karsch-Mizrachi, I., Lipman, D. J., Ostell, J., & Sayers, E. W. (2015). GenBank. *Nucleic acids research*, 44(D1), D67-D72.
- [3] Angermueller, C., Pärnamaa, T., Parts, L., & Stegle, O. (2016). "Deep learning for computational biology". *Molecular Systems Biology*.
- [4] Chen, Y., Li, Y., Narayan, R., Subramanian, A. and Xie, X. (2016), "Gene expression inference with deep learning". *Bioinformatics*, doi:10.1093/bioinformatics/btw074.
- [5] Alipanahi, B., Delong, A., Weirauch, M. T., & Frey, B. J. (2015). "Predicting the sequence specificities of DNA-and RNA-binding proteins by deep learning". *Nature Biotechnology*, 33(8), 831.
- [6] Lee, N. K., Azizan, F. L., Wong, Y. S., & Omar, N. (2018). "DeepFinder: An integration of feature-based and deep learning approach for DNA motif discovery". *Biotechnology & Biotechnological Equipment*, 1-10.
- [7] Jones, W., Alasoo, K., Fishman, D., & Parts, L. (2017). *Computational biology: deep learning. Emerging Topics in Life Sciences*, 1(3), 257-274.
- [8] Hindorff, L. A., Sethupathy, P., Junkins, H. A., Ramos, E. M., Mehta, J. P., Collins, F. S., & Manolio, T. A. (2009). "Potential etiologic and functional implications of genome-wide association loci for human diseases and traits". *Proceedings of the National Academy of Sciences*, 106(23), 9362-9367.
- [9] Quang, D., & Xie, X. (2016). "DanQ: a hybrid convolutional and recurrent deep neural network for quantifying the function of DNA sequences". *Nucleic acids research*, 44(11), e107-e107.
- [10] Besemer, J., & Borodovsky, M. (2005). "GeneMark: web software for gene finding in prokaryotes, eukaryotes, and viruses". *Nucleic acids research*, 33(suppl_2), W451-W454.

- [11] (n.d.). Retrieved from <https://study.com/academy/lesson/comparative-genomics-prokaryotes-vs-eukaryotes.html>
- [12] van Baren, M. J., & Brent, M. R. (2006). "Iterative gene prediction and pseudogene removal improve genome annotation". *Genome research*, 16(5), 678-685.
- [13] Kim, S. K., Nam, J. W., Rhee, J. K., Lee, W. J., & Zhang, B. T. (2006). "miTarget: microRNA target gene prediction using a support vector machine". *BMC Bioinformatics*, 7(1), 411.
- [14] Hoff, K. J., Tech, M., Lingner, T., Daniel, R., Morgenstern, B., & Meinicke, P. (2008). "Gene prediction in metagenomic fragments: a large scale machine learning approach". *BMC Bioinformatics*, 9(1), 217.
- [15] Borodovsky, M., Hayes, W. S., & Lukashin, A. V. (1999). "Statistical predictions of coding regions in prokaryotic genomes by using inhomogeneous Markov models". In *Organization of the prokaryotic genome* (pp. 11-33). American Society of Microbiology.
- [16] Wang Z, Chen Y, Li Y. "A Brief Review of Computational Gene Prediction Methods". *Genomics, Proteomics & Bioinformatics*. 2004;2(4):216-221. doi:10.1016/S1672-0229(04)02028-5.
- [17] Borodovsky, Mark & Mcininch, James. (1993). "GeneMark: Parallel Gene Recognition for both DNA Strands". *Computers & Chemistry*. 17. 123-133. 10.1016/0097-8485(93)85004-V.
- [18] Licon, A., Taufer, M., Leung, M.-Y., & Johnson, K. L. (2010). "A Dynamic Programming Algorithm for Finding the Optimal Segmentation of an RNA Sequence in Secondary Structure Predictions". 2nd International Conference on Bioinformatics and Computational Biology 2010, (BICoB-2010), Honolulu, Hawaii, USA, 24-26 March 2010. International Conference on Bioinformatics and Computational Biology (2nd : 2010 : Honolulu, Hawaii), 2010, 165-170.
- [19] BLAST: Basic Local Alignment Search Tool. (n.d.). Retrieved from <https://blast.ncbi.nlm.nih.gov/Blast.cgi>
- [20] Altschul, S. F., Gish, W., Miller, W., Myers, E. W., & Lipman, D. J. (1990). "Basic local alignment search tool. *Journal of molecular biology*", 215(3), 403-410.
- [21] Alipanahi, B. et al. (2015). "Predicting the sequence specificities of DNA- and RNA-binding proteins by deep learning". *Nat. Biotechnol.*, 33, 831-838.
- [22] Zhou, J. and Troyanskaya, O.G. (2015). "Predicting effects of noncoding variants with deep learning-based sequence model". *Nat. Methods*, 12, 931-934.
- [23] Quang, D., & Xie, X. (2016). "DanQ: a hybrid convolutional and recurrent deep neural network for quantifying the function of DNA sequences". *Nucleic Acids Research*, 44(11), e107. <http://doi.org/10.1093/nar/gkw226>
- [24] Yue, T., & Wang, H. (2018). "Deep Learning for Genomics: A Concise Overview". arXiv preprint arXiv:1802.00810.
- [25] Brueckner, Raymond & Schuller, Björn. (2015). "Be at Odds? - Deep and Hierarchical Neural Networks for Classification and Regression of Conflict in Speech". 10.1007/978-3-319-14081-0_19.
- [26] Hui Yu, Jing Wang, Quanhu Sheng, Qi Liu, Yu Shyr. (2018). "beRBP: binding estimation for human RNA-binding proteins", *Nucleic Acids Research*, Volume 47, Issue 5, 18 March 2019, Page e26, <https://doi.org/10.1093/nar/gky1294>.
- [27] Ritambhara Singh, Jack Lanchantin, Gabriel Robins, Yanjun Qi. (2016). "DeepChrome: deeplearning for predicting gene expression from histone modifications", *Bioinformatics*.
- [28] Mohammad Hemmat Esfe, Mohammad Hadi Hajmohammad, Nima Sina, Masoud Afrand. (2018). "Optimization of thermophysical properties of Al₂O₃/water-EG (80:20) nanofluids by NSGA-II", *Physica E: Low-dimensional Systems and Nanostructures*
- [29] What is Logistic Regression? (n.d.). Retrieved from <https://www.statisticssolutions.com/what-is-logistic-regression/>.
- [30] Ian Goodfellow, Yoshua Bengio & Aaron Courville (2016). *Deep Learning*. MIT Press.
- [31] Introduction - Vue.js. (n.d.). Retrieved from <https://vuejs.org/v2/guide/>.
- [32] JavaScript. (n.d.). Retrieved from <https://developer.mozilla.org/en-US/docs/Web/JavaScript>