Feature Selection with Ensemble Learning for Prostate Cancer Prediction from Gene Expression

Yusuf Aleshinloye Abass
Department of Computer Science
Nile University of Nigeria.

Steve A. Adeshina
Department of Computer Science
Nile University of Nigeria.

Abstract

Machine and deep learning-based models are emerging techniques that are being used to address prediction problems in biomedical data analysis. DNA sequence prediction is a critical problem that has attracted a great deal of attention in the biomedical domain. Machine and deep learning-based models have been shown to provide more accurate results when compared to conventional regression-based models. The prediction of the gene sequence that leads to cancerous diseases, such as prostate cancer, is crucial. Identifying the most important features in a gene sequence is a challenging task. Extracting the components of the gene sequence that can provide an insight into the types of mutation in the gene is of great importance as it will lead to effective drug design and the promotion of the new concept of personalised medicine. In this work, we extracted the exons in the prostate gene sequences that were used in the experiment. We built a Deep Neural Network (DNN) and Bi-directional Long-Short Term Memory (Bi-LSTM) model using a k-mer encoding for the DNA sequence and one-hot encoding for the class label. The models were evaluated using different classification metrics. Our experimental results show that DNN model prediction offers a training accuracy of 99 percent and validation accuracy of 96 percent. The bi-LSTM model also has a training accuracy of 95 percent and validation accuracy of 91 percent.

Key words: *Feature Selection, Ensemble Learning, Prostate Cancer Prediction, Gene Expression*

1. Introduction

The ability to tailor treatment to individual patients is dependent on the ability to accurately diagnose diseases and predict the disease outcomes under different treatment conditions. Omic technologies hold great promise in terms of predictors that have high throughput, but they have also been plagued by several technical challenges [1]. Omic data are known for their high dimensionality and the possibility of interaction is a huge challenge. There are large systematic source variations between experiments that are conducted in biological studies using small data samples [2]. However, recent studies on machine learning look promising and new computational methods that integrate data from many studies may help surmount existing challenges [3]. Accurate prediction of exons from Omic data will usher in a new era of molecular diagnostics [4].

Machine learning methodologies benefit from huge datasets where learning complex relationships is feasible. There is a huge number of Omic datasets in the public repository and individual biological experiments that use Omic datasets have to leverage the availability of this huge dataset. "The National Centre for Biotechnology Information (NCBI)"[1] have thousands of samples from human Ribonucleic Acid sequence (RNA-seq). These datasets cover a wide variety of tissues and diseases. Most genes do not act in isolation and using a combination of genes may prove to be more effective than using individual genes for prediction purposes.

Linear models can be used to experiment on RNA-seq data and predictors can be created from a weighted combination of gene expression values. Some of the features in the genes could reflect biological processes that are involved in multiple phenotypes; many analyses have explored the possibilities of creating complex features that incorporate biological knowledge from gene sets [5], ontology [6], or graph interaction [7]. In recent studies, unsupervised machine learning methodologies have incorporated Auto-Encoders (AE) [8], Principal Component Analysis (PCA) [9] and neural network architectures to discover such features. All of these methods are classified under representation learning and their goal is to train an unsupervised model to extract a complex feature in the model [10].

1.1 Challenges in Bioinformatics

One of the notable goals of bioinformatics is to understand the relationship between protein structure and function. To understand the structure-function relationship, one needs to know that there is a great deal

Manuscript received December 5, 2021
Manuscript revised December 20, 2021
https://doi.org/10.22937/IJCSNS.2021.21.12.73

of useful structural information that can be extracted from the primary amino acid sequence. DNA sequence prediction is very important because sequences that have similar structures also have similar functions. BLAST and FASTA [11] are two sequence alignment methods that are usually used when establishing sequence similarities. Two assumptions support the sequence similarities (it should be noted that the assumptions are valid but are not generalizable): (1) the functional element shares common sequence features; and, (2) the order of the functional elements is maintained in different sequences. Despite the various advancements in sequence alignment application methodologies, computational complexity remains a major challenge. The work of [12] which investigates the regulatory genome is an alignment-free method. The alignment-free method involves a feature extraction phase such as the special representation of DNA [13].

The advancements of deep learning architecture in terms of low-cost parallel computation has made the deep learning paradigm the choice of paradigm when dealing with big data [14] Deep learning architecture has been applied in different fields, including Natural Language Processing (NLP), Computer Vision, Speech Recognition, Voice Recognition, and Genomic Analysis [14]. It has made an effective contribution in medical science particularly in the fields of genomic medicine and medical imaging. To the best of our knowledge, only a few studies have been done in the area of prostate cancer genomic prediction. The work of [14] reviews deep learning architectures in genomic sequencing.

In this work, we explore the predictive capabilities of Deep Neural Network (DNN) and Bi-directional Long Short Term Memory (Bi-LSTM) which is a variant of Long Short Term Memory (LSTM). We use different prediction metrics to determine the predictive capabilities of the models.

This paper is organised as follows: Section 2 describes the use of deep learning architectures of DNN, LSTM, Bi-LSTM, and RNN in genomic research. We describe the building blocks of each of the deep learning architectures stated above. Section 3 presents a review of the deep learning networks for Deoxyribonucleic Acid (DNA) sequence prediction. Section 4 describes the various materials that were used in the predictive model. Section 5 captures the various deep learning model used in the experimentation. Section 6 discusses the results of the experiment in detail and provides a description of the dataset. Section 7 provides an overview of the discussion and analysis surrounding the models used in this study. Section 8 concludes the paper and presents some possible future directions for study.

2. Deep Learning Architectures

2.1 Deep Neural Network

The Artificial Neural Network (DNN) is inspired by the brain's visual input processing mechanism which takes place at multiple levels [15]. The artificial neural network emulates the processing power of large interconnected neurons inside the brain. The goal is to understand the computational theory behind the brain and the way it abstractly represents human intelligence. The Artificial Neural Network has an input layer where data enters the network; one or more hidden layers and one output layer. Every hidden layer in the ANN is made up of several neurons, where each neuron is fully connected to all the neurons in the previous layer. Each connection in the network is quantified by its weight. The weights need to be set to a suitable value, which is estimated through a training process, for the network to produce the correct output. Once the pattern is learned in the network, the network can be used to make predictions on new data, i.e. to generalise to new data. ANNs are difficult and computationally expensive to train, but they are flexible and are able to model and solve complicated problems [16]. This is why they have become a prominent method to use in machine learning and the subject of many studies. The increased use of Artificial Neural Networks is also due to the growth of big data, the availability of powerful processors for parallel computations, the ability to tweak the algorithms used in constructing and training the networks, and the development of frameworks that are easy to use. The mathematical expression for an input signal from the input layer to the hidden layer can be generalised as equations.

$$H_j^I = \sum_{j=1}^{n} \sum_{i=1}^{6} I_i w_{ij} \, I_i w_{ij} + b_j \qquad (1)$$

The output generated from the hidden layer (H_j^0), after processing by the Tansigmoidal transfer function, is shown in Equation 2.

$$H_j^0 = f\left(H_j^I\right) = \frac{2}{1+e^{-2\left(H_j^I\right)}} - 1 \qquad (2)$$

The input signal from the hidden layer to the output layer can be expressed as Equation 3 while the processing of this input signal using a transfer function to create the final output is shown as Equation 4.

$$O_k^I = \sum_{j}^{n} H_j^0 v_{jk} + b_{k/k=\,1\ to\ 2} \qquad (3)$$

$$O_k^0 = f(O_k^I) \qquad (4)$$

Where I_i – Input, w_{ij} weight of the connection between neuron of input and hidden layer.

w_{ij}- Weight of the connection between neurons of the input and hidden layer.

H_j^I -Hidden layer input

$H_j^0 - \mathit{Hidden\ layer\ output}$

v_{jk}- Weight of connection between neurons of hidden and output layer.

O_k^0 – Final output of the network

O_k^I- Input data for the output layer

b_j , b_k- Weight bias for hidden layer (j) and output layer (k)

2.2 Recurrent Neural Network

The Recurrent Neural Networks (RNN) is used for processing sequence data that evolves along the time axis. In a simpler version of RNN, the internal state h_t represents the sequence seen until the previous time step (t – 1) and it is used alongside the new input x_t.

$$h_t = \sigma(w_h x_t + u_h h_{t-1} + b_h) \qquad (5)$$

$$y_t = \sigma(w_y h_t + b_y) \qquad (6)$$

Where w_h and u_h are respectively the weight matrices for the input and the internal state, w_y is the weight matrix for producing the output from the internal state, and the two *b* are bias vectors.

RNN is known to have some drawbacks. The limitation of RNN in the above formulation is that the entire time steps have the same weight and the input contribution in the hidden state is subjected to exponential decay. A variant of RNN was introduced in [17] under the name Long Short-Term Memory (LSTM).

2.3 Long Short Term Memory (LSTM)

There are several variants of RNN that were developed to address the drawback of the simple RNN. LSTM is a popular variant of RNN. Every unit of LSTM is associated with memory that is typically referred to as a cell. In the LSTM cell unit, three gates are used to regulate the memory. The gates are the input gate (i_t), output gate (o_t), hidden state (h_t) and the forget gate(f_t). These gates help the LSTM to determine the information that needs to be added to the current call state (c_t) and the information that needs to be forgotten to update the cell state. The Equations 7 to 12 below represent the flow from the current-cell state, previous cell state, and the next state [18].

$$f_t = \sigma\left(w_f\ *\ [h_{t-1},x_t] + b_f\right) \qquad (7)$$

$$i_t = \sigma\left(w_i\ *\ [h_{t-1},x_t] + b_i\right) \qquad (8)$$

$$\tilde{c}_t = \tanh h\,(w_c\ *\ [h_{t-1},x_t] + b_c \qquad (9)$$

$$C_t = f_t * C_{t-1} + i_t * \tilde{c}_t \qquad (10)$$

$$o_t = \sigma\left(w_o\ *\ [h_{t-1},x_t] + b_o\right) \qquad (11)$$

$$h_t = o_t * tan\ h(c_t) \qquad (12)$$

The LSTM was specifically designed to overcome the challenges of the vanishing gradient problem and is deemed efficient in capturing long-term dependencies [18].

2.4 Bidirectional LSTM

The Bidirectional LSTM captures the idea that the output of recurrent unit at a time step not only depends on its past instances (past elements of the sequence) but also on future instances. This network is developed by stacking 2 layers of LSTMs over each other, thus making the output dependent on the computation of the hidden states from both the LSTM layers as opposed to one as in the unidirectional LSTM network [19]. The Bi-LSTM model computes the hidden layer $h = (h_1, \dots, h_t)$ and the output layer $o = (o_1, \dots, o_t)$ for the input sequence $s = (s_1, \dots, s_t)$, iterating through the formulas as shown in Equations 13 and 14.

$$h_t = \sigma(w_{sh} h_t + w_{hh} h_{t-1} + b_h) \qquad (13)$$

$$o_t = \sigma(w_{ho} h_t + b_o) \qquad (14)$$

The weight matrix between the input and the intermediate layer is expressed as w_{sh} , b_h is the bias vector for intermediate layer vectors and σ is the nonlinear activation function.

2.5 Softmax Layer

The RNN and LSTM need a further layer to compute a prediction task. The softmax layer [20] is composed of k units, where k is the number of different classes. Each unit is densely connected with the previous layer and computes the probability that an element is of class k using Equation 15.

$$softmax_k(x) = \frac{e^{w_k x + b_x}}{\sum_{l=1}^{k} e^{w_l x + b_l}} \qquad (15)$$

Where w_l is the weight matrix connecting the *l–th* unit to the previous later, x is the output of the previous layer and b_l is the bias for the *l–th* unit.

Softmax is widely used in deep learning as a prediction layer because of the normalised probability distribution of its outputs, which proves particularly useful during backpropagation.

2.6 Character Embedding

In this work, we evaluate models for predicting genomic sequences without providing prior information using feature engineering. One method of achieving this is represented by the use of k-mer distribution. K-mer is unique subsequences of a particular length k from larger DNA sequences. K-mer representation of biological sequence is a feature discriminant between the coding and non-coding region [21]. Another form of genomic sequence feature selection is the one-hot encoding. The one-hot encoding represents each character i of the alphabet in the sequence by a vector of length equal to the alphabet size, having all zero entries except for a single one in position i. The one-hot encoding has some notable drawbacks; a sequence that is modelled in one-hot encoding may contain limited useful information compared to other objects such as images or sound [16]. One-hot encoding cannot capture the frequency domain of features as k-mer can [22].

3. Literature Review

The RNN and LSTM were used by *DeepTarget* [23] and deep *MirGene* [24] respectively for micro ribonucleic acid (miRNA) and target prediction using expression data. Both the *DeepTarget* and *DeepMirGene* algorithms proved that miRNA can be predicted more accurately than when using a non-DL model such as *TargetScan* [25]. The DL model does not require any of the handcrafting features that are used in the other non-DL models. In performing inference on gene data expression, the authors of *D-GEX* provide a deep architecture to infer the expression of target genes from the expression available on the landmark gene [26]. The sum of 111,000 public expression profiles from Gene expression Omnibus was used by *D-GEX* which trained a multi-layer feed-forward deep neural network with three hidden layers. The results showed that the deep learning model provided better accuracy than the linear regression in inferring the expression of human genes (about 21000) based on the landmark genes (about 1000). Despite the higher accuracy of the deep learning model, when compared with other existing machine learning models, the need for improvement remains, because the model shows poor performance. The *AttentiveChrome* [27] is another variant of *DeepChrome* that was developed by the same authors of *DeepChrome* [28]. The AttentiveChrome is an LSTM model developed to enhance the capacity of the *DeepChrome* using a unified architecture to interpret dependencies among chromatin factors to control gene regulation. Tables 1 and 2 show the application of RNN and LSTM deep learning models in genomics.

***Table 1**: List of works showing the application of the RNN deep learning model in genomics*

Name	Publication	Omic Dataset	Purpose	Accuracy	Performance Gap
DeepTarget	[23]	miRNA-mRNA pairing	prediction	0.96	+25% F-measure
D-GEX	[26]	Expression of Landmark genes	Gene expression inference	An overall error of 0.3204 ± 0.0879	Outperform linear regression and KNN in most of the target genes

***Table 2**: List of works showing the application of the LSTM deep learning model in genomics*

Name	Publication	Omic Dataset	Purpose	Accuracy	Performance Gap
DeepMirGene	[24]	Positive pre-miRNA and non-miRNA	miRNA target	0.89 sensitivity	+4% f-measure
AttentiveChrome	[27]	Histone modification	Classify gene expression	AUC= 0.81	Marginally better than DeepChrome

The first approximation on how best to apply a multi-layer free-forward ANN to analyse RNA-seq gene expression data was presented in [29]. The free-forward ANN model outperformed the Least Absolute Shrinkage and Selection Operator (also known as LASSO) in analysing RNA-seq gene expression profile data. A deep learning model to predict response to therapy in cancer was implemented in [30]. The task of the training model was to perform a pharmacogenomics database of 1001 cancer cells to predict drug response. The deep learning method outperformed the current state-of-the-art machine learning frameworks for specific tasks. Table 3 shows the ANN application in genomics.

Table 3: List of works showing the application of ANN deep learning model in genomics

Name	Publication	Omic Dataset	Purpose	Accuracy	Performance Gap
DeepNet	[29]	RNA-seq	Control cases	0.7	Same or worst AUC from LASSO
DeepVariant	[30]	Cell-line with drug response	Predict-drug response	AUC=0.65	Outperform RF 0.54 AUC

4.0 Material and Methods

4.1 Data Collection

The DNA/genomic sequence of prostate cancer was obtained from NCBI which is a public nucleotide sequence database. The format of the DNA sequence data is the FASTA file. Researchers are mining the huge amount of human genome sequences that are now publicly available [31] to try to detect genetic variation with the hope of better understanding human diseases. Most genetic variations are in the form of Single-Nucleotide Polymorphisms (SNPs) and insertion/deletions of these non-synonymous SNPs are believed to be most frequently associated with disease phenotypes [32] as they may contribute pathological amino-acid substitutions or nonsense mutations in the protein product. In this research, the gene sequence that was obtained from the NCBI were tissue samples that had undergone RNA-seq. The tissue samples were from 95 human individuals representing 27 different tissues. A brief description of the prostate cancer gene sequence is given below:

- The Ataxia-Telangiectasia Mutated (ATM): The ATM gene provides instructions for making a protein that is located primarily in the nucleus of cells, where it helps control the rate at which cells grow and divide. This protein also plays an important role in the normal development and activity of several body systems, including the nervous system and immune system [33]. ATM proteins assist cells in recognising damaged or broken DNA strands. Genetic alteration in DNA repairs genes, such as ATM, has been found in prostate cancers. Mutations in the ATM gene result in an altered ATM protein that does not function normally.
- Fanconi Anaemia Complementation Group A (FANCA): The FANCA gene provides instructions for making a protein that is involved in a cell process known as Fanconi Anaemia (FA) pathway. The FA pathway is turned on (activated) when the process of making new copies of DNA, called DNA replication, is blocked due to DNA damage. The FA pathway is particularly responsible for a certain type of DNA damage known as inter-strand cross-links (ICLS). ICLS occurs when two DNA building blocks (nucleotides) on opposite strands of DNA are abnormally attached or linked together, which stops the process of DNA replication. ICLS can be caused by a build-up of toxic substances produced in the body or by treatment with certain cancer therapy drugs. Mutations in the FANCA gene are responsible for 60 to 70 percent of all cases of Fanconi anaemia [34]. This mutation changes single DNA building blocks (nucleotides) or inserts or deletes pieces of DNA in the FANCA gene. Loss of FANCA is associated with a familial history of prostate cancer [34].
- Breast Cancer Gene1 and 2 (BRCA1 & BRCA2): BRCA1 and BRCA2 are genes that encode proteins that play an important role in DNA repair. DNA may be damaged in many ways, for example, by Ultra Violet (UV) from the sunlight and exposure to other substances that cause breast or cross-links in the DNA. BRCA1 and BRCA2 are genes that were discovered in families that had a high incidence of breast cancer. In these families, the genetic alterations of BRCA1 or BRCA2 are present in the germ-line, which means that they are inherited. Inherited germ-line mutations in BRCA1 or BRCA2 greatly increase the likelihood of developing cancer of the breast or ovary as well as prostate cancer in men [35].
- Epithelial Cell Adhesion Molecule (EPCAM): The EPCAM gene provides instruction for making a protein known as an epithelial cell adhesion molecule (EPCAM). This protein is found in epithelial cells, which are cells that line the surface and cavities of the body. The EPCAM protein is found spanning the membrane that surrounds epithelial cells, where it helps cells stick to other cells (cell adhesion) [36]. EPCAM is associated with prostate growth inhibition [37].
- MutL Homolog1 (MLH1): The MLH1 gene provides instructions for making a protein that plays an essential role in repairing DNA. This protein helps fix an error that is made when DNA is copied (DNA replication) in preparation for cell division. The MLH1 protein joins with another protein known as Premenstrual Syndrome 2 (PMS2) (produced from the PMS2 gene) to form a two-protein complex called a dimer. This complex coordinates the activities of other protein that repairs errors made during DNA replication. The MLH1 gene is known as the mismatch repair gene [38].

Table 4 shows details about the six gene mutations with their annotation, the gene id, gene name, and gene. For example, ATM has a unique code in the database, also known as the accession version of NM_000051.4...17, and is a protein-coding gene.

Table 4: *Six gene mutations for prostate cancer*

Gene ID	Symbol	Gene Name	Gene Type	Scientific Name	Transcripts
472	ATM	ATM serine/ threonine kinase	proteincoding	Homo sapiens	NM_000051.4...17
2175	FANCA	FA complementation group A	proteincoding	Homo sapiens	NM_000135.4...4
672	BRCA1	BRCA1 DNA repair associated	proteincoding	Homo sapiens	NM_007294.4...6
675	BRCA2	BRCA2 DNA repair associated	proteincoding	Homo sapiens	NM_000059.4
4072	EPCAM	epithelial cell adhesion molecule	proteincoding	Homo sapiens	NM_002354.3
4292	MLH1	mutL homolog 1	proteincoding	Homo sapiens	NM_000249.4...25

4.2 Identification of Exonic Region

DNA is the most important chemical compound in living cells, viruses, and bacteria. The DNA is composed of types of different nucleotides, namely Adenine (A), Cytosine (C), Guanine (G), and Thymine (T) [39]. The coding information for protein synthesis is only carried by Q specific area of the DNA molecule called the gene [40]. The DNA is divided into gene and inter-genic species in the eukaryotic cells. The gene is further divided into exons and introns, as shown in Figure 1.

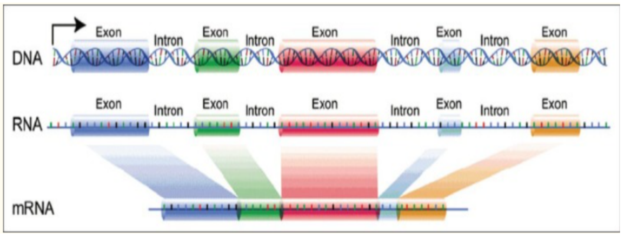


Figure 1*: Exon/Intron region for eukaryotic [41]*

There are many digital signal processing (DSP) methods presented in the literature to identify the protein-coding regions. Most importantly, these methods aim to reduce background noise in the DNA sequences, based on the Fourier technique. In [41], a Fourier technique was used to analyse the three-base periodicity in genomic sequences and was able to determine exonic regions based on the calculated power spectrum. In [42] a central frequency of $2\delta/3$ was used to remove the background noise from the DNA. The authors in [42] passed a DNA sequence through a notch filter and sliding window for Discrete Fourier Transform (DFT). In [43], a Z-curve was implemented to identify gene islands in the total DNA sequence. This is a windowless technique known as the cumulative Guanine-Cytosine (GC)-profile method. The main feature of this method is that the resolution of the algorithm output displayed in the genomic GC content is very high because no sliding window is required, but the downside is that the computational complexity of the method is also very high. Identifying protein-coding regions (exons) has helped genetic engineers to isolate proteins. This has also helped to tailor personalised drugs for various diseases. Predicting protein-coding regions in the DNA sequence is a vital step in understanding genetic processes [44].

In this work, the candidate exons from the gene sequence mentioned in Section 4.1 are extracted from the DNA sequence as shown in Figure 2 below:

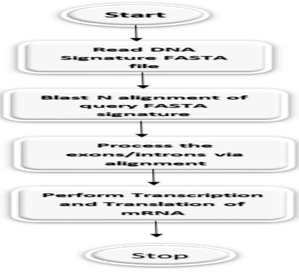
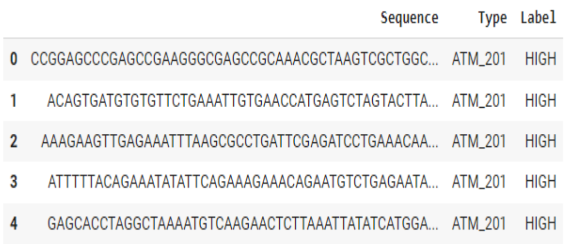

Start
Read DNA Signature FASTA file
Blast N alignment of query FASTA signature
Process the exons/introns via alignment
Perform Transcription and Translation of mRNA
Stop

Figure 2*: Extraction of exons from FASTA file*

The process of sequencing individual genes is usually performed at the exon level. First, the FASTA file containing the DNA sequence is read based on the gene of interest and this is obtained from the sequence database, as discussed in Section 4.1. Next, the corresponding genomic sequence is identified and retrieved. The exon/intron is determined once the genomic and mRNA sequences are obtained. The section of the mRNA is transcripted, the section of the mRNA that does not code for proteins is removed and the section that codes for protein are combined to a long chain of mRNA. Finally, the long chain of mRNA is translated.

The exonic dataset is then labelled based on genetic type. Figure 3 shows a sample dataset with a genomic sequence and class label.

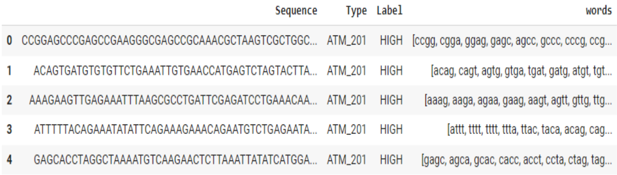


	Sequence	Type	Label
0	CCGGAGCCCGAGCCGAAGGGCGAGCCGCAAACGCTAAGTCGCTGGC...	ATM_201	HIGH
1	ACAGTGATGTGTGTTCTGAAATTGTGAACCATGAGTCTAGTACTTA...	ATM_201	HIGH
2	AAAGAAGTTGAGAAATTTAAGCGCCTGATTCGAGATCCTGAAACAA...	ATM_201	HIGH
3	ATTTTTACAGAAATATATTCAGAAAGAAACAGAATGTCTGAGAATA...	ATM_201	HIGH
4	GAGCACCTAGGCTAAAATGTCAAGAACTCTTAAATTATATCATGGA...	ATM_201	HIGH

***Figure 3**: Genomic sequences, types, and labels.*

4.3 Data Preprocessing

The task of preprocessing data is one of the most critical steps in most machine learning and deep learning algorithms that involve numerical data rather than categorical data. There are many techniques available to convert categorical data to numerical data. An encoding technique is a process of converting the categorical data of the nucleotide into numerical form. In this paper, label encoding and k-mer encoding are used to encode the DNA sequence. Each input DNA sequence label was converted into a matrix n · l by one-hot encoding, where n corresponds to the three labels High, Low, and Normal represented by binary vectors High = [1,0,0,0], Low = [0,1,0,0], and Normal = [0,0,0,1], respectively, and the length l equals 4 which represents the length of the k-mer. In the k-mer encoding technique, the DNA sequence is converted into a bag of words that resemble English words. Each DNA sequence in the dataset is transformed into a k-mer of size k as shown in Figure 4 and all k-mers generated are concatenated to form a sentence.



	Sequence	Type	Label	words
0	CCGGAGCCCGAGCCGAAGGGCGAGCCGCAAACGCTAAGTCGCTGGC...	ATM_201	HIGH	[ccgg, cgga, ggag, gagc, agcc, gccc, cccg, ccg...
1	ACAGTGATGTGTGTTCTGAAATTGTGAACCATGAGTCTAGTACTTA...	ATM_201	HIGH	[acag, cagt, agtg, gtga, tgat, gatg, atgt, tgt...
2	AAAGAAGTTGAGAAATTTAAGCGCCTGATTCGAGATCCTGAAACAA...	ATM_201	HIGH	[aaag, aaga, agaa, gaag, aagt, agtt, gttg, ttg...
3	ATTTTTACAGAAATATATTCAGAAAGAAACAGAATGTCTGAGAATA...	ATM_201	HIGH	[attt, tttt, tttt, ttta, ttac, taca, acag, cag...
4	GAGCACCTAGGCTAAAATGTCAAGAACTCTTAAATTATATCATGGA...	ATM_201	HIGH	[gagc, agca, gcac, cacc, acct, ccta, ctag, tag...

***Figure 4**: K-mer encoding of DNA sequence*

5. Classification Models

Given an exonic sequence of length L_0, we split the sequence into k-mer of size k. We extract all subsequences of length k resulting in a k-mer sequence with length $L = (L_0 - k) + 1$. In this work, the two prediction models are DNN and BLSTM. The task of preserving the nucleotide information of the DNA sequence is based on the use of label encoding and k-mer techniques. The k-mer represents the features and is split into training and test sets. The workflow for the experimentation is shown in Figure 5.

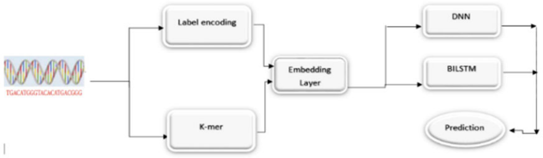


***Figure 5:** DNA sequence prediction model*

5.1 Deep Neural Network

We built a sequential model that had four layers: the input layer, activation layer, dropout layer, and output layer. The network layer was embedded with a vocabulary size of 12,850. A hidden layer of 50 neurons was added, specifically an output layer that is a dense layer with 3 outputs (multiclass prediction) and a sigmoid activation function. The dropout layer rate was set to 0.5, which prevented the model from overfitting. Table 5 presents a summary of the characteristics of the DNN deep learning model. Once the model was prepared, it was enhanced with loss = categorical_crossentropy, an Adam optimiser, and accuracy metrics to evaluate its predictions.

***Table 5:** Hyperparameters used in the DNN prediction model*

Hyperparameters	Values
Epochs	20
Batch size	5
Architecture Function	Softmax
Training size	80%
Validation	20%
Learning rate	0.0001

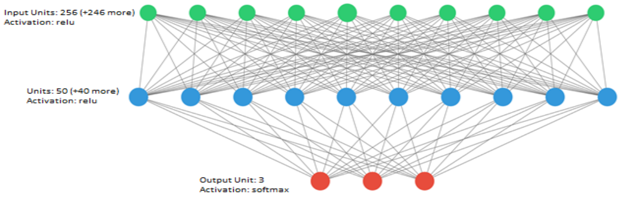


***Figure 6:** Implemented DNN Architecture*

5.2 Long Short-Term Memory (LSTM) and Bi-directional LSTM

The LSTM is a recurrent neural network (RNN) that can learn long-term dependencies in a sequence and it is also used in sequence prediction. The LSTM is made up of memory blocks called cells. Each of the cells comprises three gates: input, output, and forget. The LSTM is built to remember and forget things [45]. Figure 7 depicts the basic LSTM model's overall architecture. It is capable of learning the input sequence which is the k-mer of sequences and recognising the long sequence. The current state of the LSTM is calculated using Equation 16.

$$h_t = f(h_{t-1},X_t) \qquad (16)$$

Where X_t the input is state, h_t is the current state and h_{t-1} is the previous state.

In the LSTM, information is removed from the cell state by the forget gate. When information becomes invalid for the prediction process, the forget gate outputs a value of 0, indicating that the data should be removed from the memory cell. This gate takes two inputs: h_{t-1} (input from the previous state) and X_t (input from the current state). The input is multiplied by a weight and then added with bias. Finally, the *sigmoid* function is applied. The value output by the sigmoid function is within the range 0 to 1. The input gate in the LSTM is responsible for adding all the relevant values to the cell state. It involves the activation functions. Initially, the *sigmoid* function controls what value is added to the cell state. Secondly, the *tanh* function returns values in the range of -1 to 1, indicating all possible values applied to the cell state. The output gate task is to decide the value that can be in the output by employing the sigmoid activation function and tanh activation function to the cell state. LSTM layer extracted memory units are added after feature extraction for prediction purposes. The feature extracted is then given as an input to the LSTM layer for prediction. The LSTM model includes a dropout layer and regulation technique to reduce the overfitting problem.

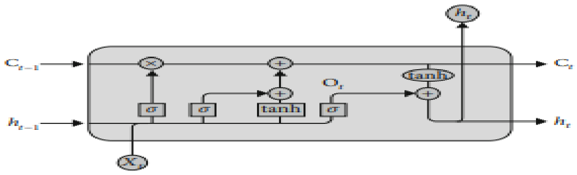


Figure 7: Architecture of the LSTM model.

In this work, the bi-directional LSTM which is a variant of the LSTM model, is used for DNA sequence prediction. The model uses k-mer feature extraction and bi-directional LSTM for prediction. The bi-directional LSTM contains two RNN, one to learn the dependencies in the forward sequence and another to learn the dependencies in the backward sequence [46]. Table 6 provides a summary of the bi-LSTM deep learning model. Once the model was prepared, it was enhanced with loss= categorical_crossentropy, an Adam optimiser, and accuracy metrics to evaluate its predictions. The architecture of the bi-directional LSTM is given in Figure 8.

Table 6: Hyperparameters used in the Bi-LSTM prediction model

Hyperparameters	Values
Epochs	20
Batch size	32
Architecture Function	Softmax
Training size	80%
Validation	20%
Learning rate	0.0001

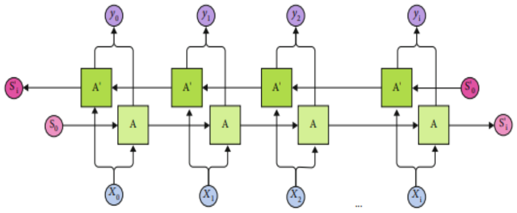


Figure 8: Architecture of Bi-directional LSTM

5.3 Model Training

The input sequence, as shown in Figure 4, is given as input to the sequence models. Each cell takes information as input from the previous cell in the form of a hidden state vector and cell state vector. The output is the concatenation of the hidden and cell-state vectors. A softmax function is applied to obtain probability distribution. To reduce overfitting, early stopping is used to end training when the validation loss does not decrease.

5.4 Model Evaluation

The deep learning models (DNN and Bi-LSTM) were used to simulate the sequence of prostate cancer gene sequence prediction. The models were all used as described in previous sections. A count vectorizer was used to convert the bag of words (k-mer sequences) into a vector space based on the word frequency. The labels were all encoded using a one-hot encoder. The sentence *s* is passed through the hidden state and the cell state from the input layer. For the models to make a prediction, the output has to be passed through the dense layer. The program architecture for DNN and Bi-LSTM models is shown in Figures 9 and 10, respectively.

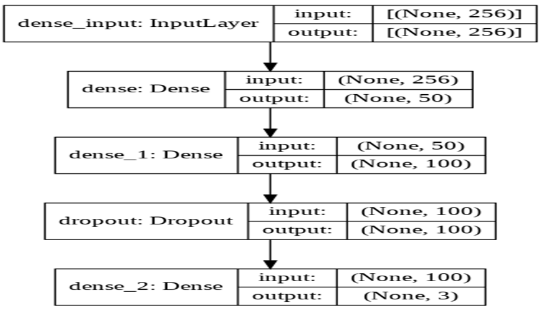


***Figure 9**: DNN Program Architecture*

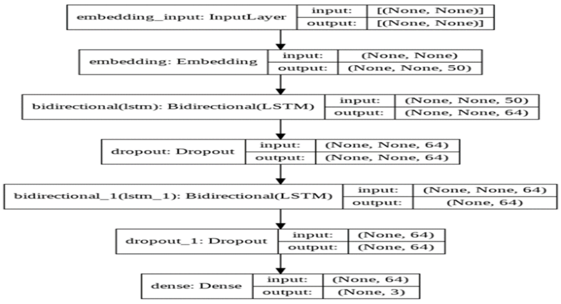


***Figure 10:** Bi-LSTM Program Architecture*

6. Results

The deep learning models experimented with a system that has a configuration of 8000 MB of RAM. The dataset consists of 13,439 inputs and was divided into training and testing set with a ratio of 80 percent and 20 percent, respectively. The training set consisted of 10,751 and the testing set consisted of 2,687 samples. The maximum sequence length was 4 and the vocabulary size was 256. In the training phase, the categorical cross-entropy function was used as a loss function. The goal of the loss function is to calculate the error between the actual output and the target label, for which the training and update of the weight are done. The implemented models were all tested by varying the hyper-parameters, such as the number of the epoch, number of layers, and embedding dimensions. The classification models were all evaluated using different classification metrics such as accuracy, precision, recall, and F1 score. The classification metrics were all calculated from the confusion metric by obtaining the True Positive Gene (TPGene), True Negative Gene (TNGene), False Positive Gene (FPGene), and False Negative Gene (FNGene). The formulae for each of the stated metrics are given below. Figures 11 and 12 show the confusion metrics of the DNN and Bi-LSTM, respectively. Tables 7, 8, 9, and 10 show the computer values for the classification metrics for both models. Figures 13 and 14 show the training and validation accuracy and the training and validation loss for both models respectively.

$$Accuracy = \frac{\text{TPGene+TNGene}}{TPGene+TNGene+FPGene+FNGene}$$

$$Specificity = \frac{\text{TNGene}}{TNGene + FPGene}$$

$$Sencitivity = \frac{\text{TNGene}}{TPGene + FNGene}$$

$$Precision = \frac{\text{TPGene}}{TPGene + FPGene}$$

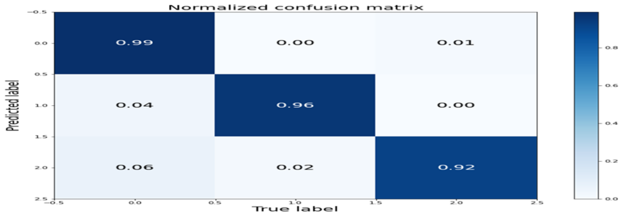


Figure 1: DNN Confusion Matrix

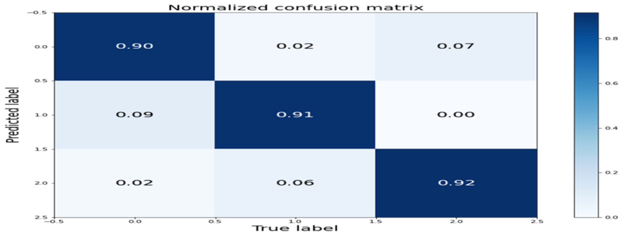


***Figure 2**: Bi-LSTM Confusion Matrix*

Table 6: DNN Training metrics

	Labels		
	High	**Low**	**Normal**
Precision	**0.98**	**0.99**	**0.98**
Recall	**0.99**	**0.98**	**0.99**
F1-Score	**0.98**	**0.99**	**0.99**
Training Accuracy			**0.99**

***Table 7:** DNN Validation Metrics*

	Labels		
	High	**Low**	**Normal**
Precision	**0.93**	**0.99**	**0.96**
Recall	**0.99**	**0.96**	**0.97**
F1-Score	**0.98**	**0.92**	**0.95**

Test Accuracy			0.96

Table 8: Bi-LSTM Training Metrics

	Labels		
	High	Low	Normal
Precision	0.91	1	0.95
Recall	0.97	0.91	1
F1-Score	0.94	0.95	0.97
Training Accuracy			0.95

Table 9: Bi-LSTM Test Metrics

	Labels		
	High	Low	Normal
Precision	0.9	0.93	0.88
Recall	0.9	0.91	0.92
F1-Score	0.9	0.92	0.9
Test Accuracy			0.91

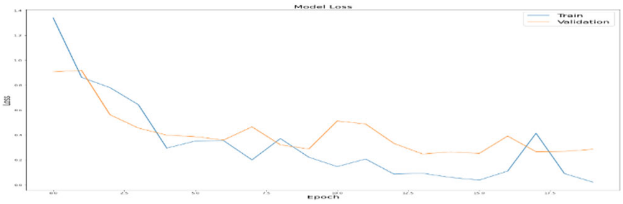


Figure 3: DNN training and validation loss

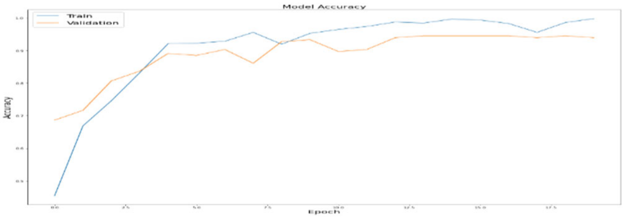


Figure 4: DNN training and validation accuracy

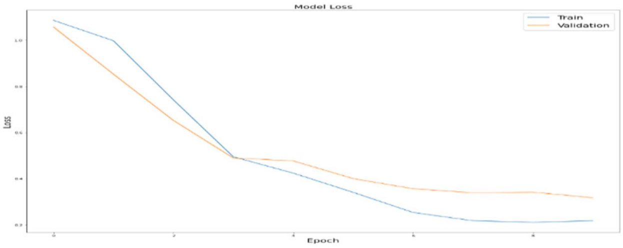


Figure 5: Bi-LSTM Training and validation Loss

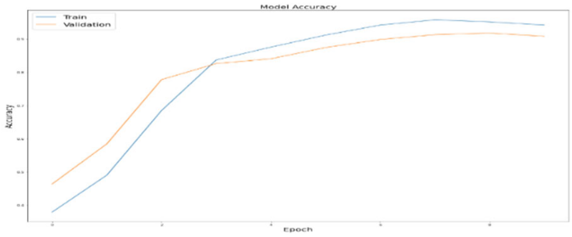


Figure 6: Bi-LSTM training and validation accuracy

7. Discussion and Analysis

The importance given to each of the stated metric evaluations differs across domains in which they are applicable. In data science, a data scientist will look at precision and recall to evaluate the built model. In the field of medicine, specificity and sensitivity are used to evaluate the medical test. In real-life applications, the two are very similar, but different. Given the sets of positive sequences, sensitivity is the set of the sequence that is defined by the positive model. If the model predicts very high sensitivity, there is a chance that a few positive cases of the sequence are expected to be false-positive. The average for the recall and precision values is the F1-score. Precision is the percentage of positive words (k-mers) identified by the model in the sequence and the specificity values are derived from how well the model determines the negative cases in the sequence. The superiority of bi-LSTM is based on the fact that it has been reported in [47] that the use of bi-LSTM outperforms the LSTM model, despite the higher accuracy recorded by the DNN in both training and validation sets. The DNN suffers from vanishing and exploding gradients. The DNN model cannot provide a clear-cut relationship among the interconnected parameters of the exons prediction task. The prediction task is one of the more useful tasks when it comes to personalised medicine. DNA sequence prediction provides experts in bioinformatics and medicine the opportunity to explore variations in gene mutation.

8. Conclusion

The rapid advances in genomic technology have presented new opportunities for medical experts and biotechnologists. The advancement in life science technologies also brings challenges. The application of various deep learning models in bioinformatics and medicine poses several challenges. Moreover, the interdisciplinary approach has led to the development of better models that can be used in prediction tasks. In this work, we have discussed the characteristics of some notable deep learning models for computer vision.

An exon was extracted from the dataset by way of preprocessing it. Exons are proteins that carry information. The datasets were obtained from the National Centre for Biotechnology Information (NCBI). The dataset was encoded and tested using both one-hot encoding and k-mer encoding. We used a DNN and bi-LSTM to train and test it. The dataset could only be evaluated with accuracy metrics. The models used has higher accuracies, and the DNN model's limitations when dealing with sequence datasets were highlighted. In both the training and test sets of models, the recall values were high, indicating that the model was highly sensitive when it came to identifying the k-mers, which are exons. In the future, more than one model will be combined to predict prostate cancer DNA sequences.

Conflicts of Interest

The authors declare they have no conflict of interest.

Authors' Contribution

Y. A. Abass wrote the manuscript with contributions from the named co-authors.

Acknowledgment

The authors thank the National Information Technology Development Agency (NITDA) and Nile University of Nigeria (NUN) for supporting Y.A. Abass's studies in Nigeria. The authors thank the anonymous reviewers for their objective remarks and their suggestions on the paper.

References

[1] N. S. Madhukar and O. Elemento, "Bioinformatics approaches to predict drug responses from genomic sequencing," *Cancer Systems Biology,* p. 277–296, 2018.

[2] S. Li, P. P. Łabaj, P. Zumbo, P. Sykacek, W. Shi, L. Shi, J. Phan, P.-Y. Wu, M. Wang, C. Wang and others, "Detecting and correcting systematic variation in large-scale RNA sequencing data," *Nature biotechnology,* vol. 32, p. 888–895, 2014.

[3] Y. A. Abass and S. A. Adeshina, "Deep Learning Methodologies for Genomic Data Prediction," *Journal of Artificial Intelligence for Medical Sciences,* 2021.

[4] P. Mamoshina, A. Vieira, E. Putin and A. Zhavoronkov, "Applications of deep learning in biomedicine," *Molecular pharmaceutics,* vol. 13, p. 1445–1454, 2016.

[5] A. Subramanian, P. Tamayo, V. K. Mootha, S. Mukherjee, B. L. Ebert, M. A. Gillette, A. Paulovich, S. L. Pomeroy, T. R. Golub, E. S. Lander and others, "Gene set enrichment analysis: a knowledge-based approach for interpreting genome-wide expression profiles," *Proceedings of the National Academy of Sciences,* vol. 102, p. 15545–15550, 2005.

[6] A. Arbaaeen and A. Shah, "Ontology-Based Approach to Semantically Enhanced Question Answering for Closed Domain: A Review," *Information,* vol. 12, p. 200, 2021.

[7] K. Zarringhalam, D. Degras, C. Brockel and D. Ziemek, "Robust phenotype prediction from gene expression data using differential shrinkage of co-regulated genes," *Scientific reports,* vol. 8, p. 1–10, 2018.

[8] R. Lopez, J. Regier, M. B. Cole, M. I. Jordan and N. Yosef, "Deep generative modeling for single-cell transcriptomics," *Nature methods,* vol. 15, p. 1053–1058, 2018.

[9] Y.-J. Shen and S.-G. Huang, "Improve survival prediction using principal components of gene expression data," *Genomics, proteomics & bioinformatics,* vol. 4, p. 110–119, 2006.

[10] Y. Bengio, A. Courville and P. Vincent, "Representation learning: A review and new perspectives," *IEEE transactions on pattern analysis and machine intelligence,* vol. 35, p. 1798–1828, 2013.

[11] R. C. Edgar, "Search and clustering orders of magnitude faster than BLAST," *Bioinformatics,* vol. 26, p. 2460–2461, 2010.

[12] L. Pinello, G. Lo Bosco and G.-C. Yuan, "Applications of alignment-free methods in epigenomics," *Briefings in Bioinformatics,* vol. 15, p. 419–430, 2014.

[13] G. L. Bosco, "Alignment free dissimilarities for nucleosome classification," in *International Meeting on Computational Intelligence Methods for Bioinformatics and Biostatistics*, 2015.

[14] T. Yue and H. Wang, "Deep learning for genomics: A concise overview," *arXiv preprint arXiv:1802.00810,* 2018.

[15] J. Schmidhuber, "Deep learning in neural networks: An overview," *Neural networks,* vol. 61, p. 85–117, 2015.

[16] A. Krizhevsky, I. Sutskever and G. E. Hinton, "ImageNet classification with deep convolutional neural networks," *Communications of the ACM,* vol. 60, p. 84–90, 2017.

[17] I. Goodfellow, Y. Bengio and A. Courville, Deep learning, MIT press, 2016.

[18] C. Olah, “Understanding lstm networks–colah’s blog,” *Colah. github. io,* 2015.

[19] H. P. Desai, A. P. Parameshwaran, R. Sunderraman and M. Weeks, “Comparative study using neural networks for 16S ribosomal gene classification,” *Journal of Computational Biology,* vol. 27, p. 248–258, 2020.

[20] J. S. Bridle, “Probabilistic interpretation of feedforward classification network outputs, with relationships to statistical pattern recognition,” in *Neurocomputing*, Springer, 1990, p. 227–236.

[21] M. Axelson-Fisk, “Comparative Gene Finding,” in *Comparative Gene Finding*, Springer, 2010, p. 157–180.

[22] L. Fu, Q. Peng and L. Chai, “Predicting dna methylation states with hybrid information based deep-learning model,” *IEEE/ACM transactions on computational biology and bioinformatics,* vol. 17, p. 1721–1728, 2019.

[23] B. Lee, J. Baek, S. Park and S. Yoon, “deepTarget: end-to-end learning framework for microRNA target prediction using deep recurrent neural networks,” in *Proceedings of the 7th ACM international conference on bioinformatics, computational biology, and health informatics*, 2016.

[24] S. Park, S. Min, H. Choi and S. Yoon, “deepMiRGene: Deep neural network based precursor microrna prediction,” *arXiv preprint arXiv:1605.00017,* 2016.

[25] B. P. Lewis, I.-h. Shih, M. W. Jones-Rhoades, D. P. Bartel and C. B. Burge, “Prediction of mammalian microRNA targets,” *Cell,* vol. 115, p. 787–798, 2003.

[26] Y. Chen, Y. Li, R. Narayan, A. Subramanian and X. Xie, “Gene expression inference with deep learning,” *Bioinformatics,* vol. 32, p. 1832–1839, 2016.

[27] J. Lanchantin, R. Singh, B. Wang and Y. Qi, “Deep motif dashboard: Visualizing and understanding genomic sequences using deep neural networks,” in *Pacific Symposium on Biocomputing 2017*, 2017.

[28] R. Singh, J. Lanchantin, G. Robins and Y. Qi, “DeepChrome: deep-learning for predicting gene expression from histone modifications,” *Bioinformatics,* vol. 32, p. i639–i648, 2016.

[29] D. Urda, J. Montes-Torres, F. Moreno, L. Franco and J. M. Jerez, “Deep learning to analyze RNA-seq gene expression data,” in *International work-conference on artificial neural networks*, 2017.

[30] B. M. Kuenzi, J. Park, S. H. Fong, K. S. Sanchez, J. Lee, J. F. Kreisberg, J. Ma and T. Ideker, “Predicting drug response and synergy using a deep learning model of human cancer cells,” *Cancer cell,* vol. 38, p. 672–684, 2020.

[31] E. S. Lander, L. M. Linton, B. Birren, C. Nusbaum, M. C. Zody, J. Baldwin, K. Devon, K. Dewar, M. Doyle, W. FitzHugh and others, “Initial sequencing and analysis of the human genome,” 2001.

[32] S. Sunyaev, J. Hanke, A. Aydin, U. Wirkner, I. Zastrow, J. Reich and P. Bork, “Prediction of nonsynonymous single nucleotide polymorphisms in human disease-associated genes,” *Journal of molecular medicine,* vol. 77, p. 754–760, 1999.

[33] Y. Miura, Y. Sakurai and T. Endo, “O-GlcNAc modification affects the ATM-mediated DNA damage response,” *Biochimica et Biophysica Acta (BBA)-General Subjects,* vol. 1820, p. 1678–1685, 2012.

[34] C. L. M. Marcelis and A. P. M. de Brouwer, “Feingold syndrome 1,” 2019.

[35] E. Castro and R. Eeles, “The role of BRCA1 and BRCA2 in prostate cancer,” *Asian journal of andrology,* vol. 14, p. 409, 2012.

[36] K. Tutlewska, J. Lubinski and G. Kurzawski, “Germline deletions in the EPCAM gene as a cause of Lynch syndrome–literature review,” *Hereditary cancer in clinical practice,* vol. 11, p. 1–9, 2013.

[37] J. Ni, P. Cozzi, J. Beretov, W. Duan, J. Bucci, P. Graham and Y. Li, “Epithelial cell adhesion molecule (EpCAM) is involved in prostate cancer chemotherapy/radiotherapy response in vivo,” *BMC cancer,* vol. 18, p. 1–12, 2018.

[38] D. E. Beaudoin, N. Longo, R. A. Logan, J. P. Jones and J. A. Mitchell, “Using information prescriptions to refer patients with metabolic conditions to the Genetics Home Reference website,” *Journal of the Medical Library Association: JMLA,* vol. 99, p. 70, 2011.

[39] D. Anastassiou, “Genomic signal processing,” *IEEE signal processing magazine,* vol. 18, p. 8–20, 2001.

[40] S. Tiwari, S. Ramachandran, A. Bhattacharya, S. Bhattacharya and R. Ramaswamy, “Prediction of probable genes by Fourier analysis of genomic sequences,” *Bioinformatics,* vol. 13, p. 263–270, 1997.

[41] H. a. S. M. a. S. M. a. G. F. Saberkari, “Prediction of protein coding regions in DNA sequences using signal processing methods,” in *2012 IEEE Symposium on Industrial Electronics and Applications*, 2012.

[42] H. Saberkari, M. Shamsi and M. H. Sedaaghi, “Identification of genomic islands in DNA sequences using a non-DSP technique based on the Z-Curve,” in *11th Iranian Conference on Intelligent Systems (ICIS 2013) February 27th & 28th*, 2013.

[43] S. S. Sahu, “Analysis of Genomic and Proteomic Signals Using Signal Processing and Soft Computing Techniques,” 2011.

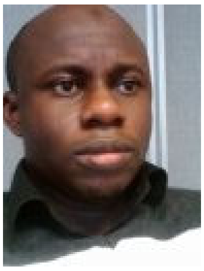
[44] G. De Clercq, “DEEP LEARNING FOR CLASSIFICATION OF DNA FUNCTIONAL SEQUENCES,” 2019.

[45] N. Mughees, S. A. Mohsin, A. Mughees and A. Mughees, “Deep sequence to sequence Bi-LSTM neural networks for day-ahead peak load forecasting,” *Expert Systems with Applications,* vol. 175, p. 114844, 2021.

[46] S. Siami-Namini, N. Tavakoli and A. S. Namin, “A comparative analysis of forecasting financial time series using arima, lstm, and bilstm,” *arXiv preprint arXiv:1911.09512,* 2019.

[47] D. P. Snustad and M. J. Simmons, Principles of genetics, John Wiley & Sons, 2015.

Steve A. Adeshina is currently a Professor of Computer Engineering and Vision at the Nile University of Nigeria (NUN) where he also serves as Dean of the Faculty of Engineering. Additionally, he was a Deputy Vice Chancellor (Administration). He graduated with a B. Eng. degree in Electrical and Electronics Engineering from the University of Ilorin, Nigeria and a PhD in Computer Vision from the University of Manchester, Manchester, UK. After a very successful computing career in the private and public sector in Nigeria, he joined the Nile University of Nigeria and has served in several academic and administrative positions at NUN. His research interest is in using Computer vision techniques in analyzing biomedical images. He is currently working on using deep learning techniques for automated detected of cancer in different medical images. He has published several works in this respect. Additionally, he has some interest in the use of technology in achieving good governance through electronic voting.



Yusuf A. Aleshinloye received his BSc. (Hons.) degree in Computer Science from the University of Abuja, Abuja, Nigeria in 2010 and the MSc. degree in Advanced Computer Science with specialisation in Artificial Intelligence from the University of Manchester, United Kingdom, 2012. He is currently pursuing a PhD. degree in Computer Science at Nile University of Nigeria, Abuja, Nigeria. In 2015, he joined the Nile University of Nigeria as a lecturer in the Department of Computer Science and a member of the Faculty of Science at the University. His area of research interest is to investigate how deep learning methodologies can be used for prediction purposes in Computer Vision.