Received: October 11, 2022. Revised: November 17, 2022.

International Journal of
Intelligent Engineering & Systems

http://www.inass.org/

A New Technique for Genetic Mutations Detection and Classification Using Deep Learning

Rana H. Saloom[1]* **Hussein K. Khafaji[2]**

[1]*Informatics Institute for Postgraduate Studies (IIPS), Iraqi Commission for Computers and Informatics (ICCI) Baghdad, Iraq*
[2]*Computer Communications Engineering Department, Al-Rafidain University College Baghdad, Iraq*
* Corresponding author's Email: Phd202020565@iips.icci.edu.iq

Abstract: Numerous variables, such as changes in the environment, toxins, spontaneous mutations, and replication mistakes, can have an impact on DNA. It is known as a gene mutation when this occurrence permanently alters the DNA sequence that forms a gene, changing it from the sequence seen in most people. Alleles, which are small variations within the same gene, are produced as a result of mutations. Each person is different due to these minute variations in their DNA sequence. In particular, some mutations affect just the carriers, whilst others affect both all children and the carrier organism's progeny. Changes in the DNA sequence (genetic mutations) are one of the reasons that lead to life-threatening disorders such as cancer and other diseases, so it has become necessary to detect these mutations early and know their types and their impact on the DNA sequence The DNA modifications developing in the cells of the following generation are what bioinformatics is most concerned about. In this paper, an efficient approach based on the architecture of long-term memory (LSTM) recurrent neural networks is presented. The method involves locating mutations using the Needleman algorithm that compares the reference DNA sequence with the mutant sequence. The deep neural network is then trained on the Cancer Cell Lines portal (CCLE) dataset to classify the different types of genetic mutations that have been identified. The algorithm reports the mutation type and locus for each pair of DNA inputs. Finally, the simulation output demonstrates the effectiveness of the algorithm. By testing the method, it was found that it was able to identify the mutation type with 100% accuracy.

Keywords: Bioinformatics, DNA sequence, DNA mutation, Sequence alignment, Mutation classification.

1. Introduction

All living things have genes, a unit of heredity, which store their genetic information. The way those genes are expressed in a given cell defines that cell's activity. As a result, any kind of gene mutation, whether single or several, can cause an imbalance in the expression of genes, which is known as a genetic illness [1].The somatic mutations can cause a variety of harmful biological effects, but DNA is particularly vulnerable to somatic mutations due to its greater susceptibility to them [2, 3] some of these mutations have been linked to critical roles in the development and metastasis of cancer [4]. Scientists working in bioinformatics still struggle with accurately predicting mutation subtypes [5] Numerous machine-learning techniques are now being used to tackle biological issues.However, it is quite challenging to reliably identify somatic mutations from the vast sequencing data. Numerous scholars have been attempting to solve this issue in recent years. Identifying the many subtypes of mutations is very important since doing so can aid in the development of a system for detecting diseases in their early stages by identifying the mutation that causes those diseases [6].

In bioinformatics, pairwise sequence alignment (PWSA) is one of the crucial jobs. In order to identify areas of similarity, a pair of genetic sequences (such as DNA and RNA) with different characteristics are arranged. The goal is to find the alignment that has the highest overall score, or the most base-to-base matches, without changing the order of the bases in

either sequence. Gap-to-gap pairings are also not allowed [7]. A pairwise sequence alignment technique based on dynamic programming, the Needleman-Wunsch (NW) algorithm is primarily used to find the best global alignment between two biological sequences (such as DNA, RNA, and protein) in O(MN) time and space where N is the length of reference sequences and M is the length of query sequences Matrix initialization, similarity score computation, traceback, and result generation are the four stages of the NW algorithm [8]. Classification algorithms in data mining can identify a certain pattern in a vast amount of data. By using both features created from the existing data and attributes that are presented in the data.

The contribution of this paper is to detect the genetic mutation in the DNA sequencing and its location and then determine the main category of the mutation (deletion, substitution, insertion) and the detection of this mutation subclass (Missense substitution, Nonsense substitution, Silent substitution, Read-through substitution, Frameshift insertion/deletion) by comparing the reference sequence with the mutant sequence using pairwise sequence alignment (PWSA) to align the mutation-free DNA sequence with the mutant sequence based on Needleman algorithm, then encoding the reference and mutant sequences and dividing them into amino acids to be classified and to know the main and subtype of the mutation. The results of the technique used in this paper outperformed the results achieved in the previous studies with an accuracy of 100%.

The remainder of this research is structured as follows: Previous studies that dealt with the identification of genetic mutations and their types and the theoretical foundations of this research will be explained in the next section. The study methodology is explained in the third part, the analysis and discussion of the results are explained in the fourth section, and the conclusions are explained in the last section.

2. Related works

In this section, we will discuss prior research on the DNA mutations classification:

In 2017, Jinliang Yanget al. utilized a program to compare and analyze the symbols of amino acids along the polypeptide chain in order to model and discover mutation categories and kinds using the PetriNet network. The model is helpful for determining if protein function and structure are impacted by mutations [9].

A strategy for classifying genetic variants based on clinical evidence was provided by Resham N. Waykole et al. in 2018. They extracted characteristics from genes and their variations using a single hot coding strategy in order to detect a mutation. The clinical transcript data were processed using the tf-idf algorithm to extract features. Logistic regression is used for classification. The findings indicate a 64% [10] classification accuracy.

In 2019, Jingwen Xuet al.suggested a classification scheme for genetic mutations based on empirical data. The approach combines a repeating unit with a bidirectional sequential neural network and a convolutional neural network. With an accuracy of 84%, the model demonstrated its superiority over the conventional single neural network model [11].

Vikalp Kumar Singh presented a project to treat hemophilia in 2020. 7,784 patient mutations were found using the position specific mutation (PSM) and one hot coding (OHE) techniques to predict the severity of the disease. Different ML algorithms were examined and trained to categorize the mutation severity level from the dataset processed by PSM and OHE approaches. Surprisingly, PSM beat OHE in terms of accuracy and efficiency of training and prediction time, with improvements in these areas of 91%. The use of PSM with various ML algorithms further increased the accuracy of risk prediction [12].

In 2021, Samuel Peacocketal.suggested a machine learning approach to automatically classify mutation by developing and training a recurrent neural network of an abstract syntax tree. The experimental analysis of 582 mutations revealed that this approach had a 90% accuracy rate for classifying mutations [13].

Natural Language Processing (NLP) methods were proposed by Meenu Gupta et al. in 2012 to categorize the genetic alterations based on clinical data. For Text conversion into a matrix of token counts, three models for text transformation (CountVectorizer, TfidfVectorizer, and Word2Vec)are used. The sparse matrix of text descriptions is subjected to the application of three machine learning classification models, namely Logistic Regression, Random Forest, and XGBoost, as well as the Recurrent Neural Network (RNN) model of deep learning. The suggested classifiers' accuracy score is assessed by 70% [14].

Using a 1-D convolutional neural network (1D-CNN), a Bidirectional Long Short-Term Memory (BiLSTM), and Bidirectional Gated Recurrent Unit (Bi-GRU), UntariNoviaWisestyetal.presented the sequential labeling model in 2022 as a method to concurrently identify type and index alterations of DNA sequences. The suggested model uses Bi-GRU and BiLSTM to report F1 scores of 0.9596 [15].

3. LSTM

A specific kind of RNN called Long Short-Term Memory (LSTM) networks is able to learn long-term dependencies. These networks perform amazingly effectively for avariety of issues. LSTMs are especially made to avoid the issue of long-term reliance [16]. LSTMs achieve this goal by naturally retaining information for lengthy periods of time. A sequence of repeatedly used neural network modules is the shape that all recurrent neural networks have. This repeating module in conventional RNNs has a fairly straightforward structure that may be a straightforward activation layer with the hyperbolic tangent function [17]. LSTM networks, on the other hand, also feature a similar sequence structure, but the repeating module is built differently. It contains four interacting layers, as shown in Fig. 1, as opposed to just one layer of a single neural network [18].

Lengthy Short-Term Memory (LSTM) neural networks are able to cope with the issue of long time-dependent inputs and solve the vanishing and expanding gradients concerns [19]. LSTMs employ the notion of gates to simplify and effectively perform computations using both Long Term Memory (LTM) and Short Term Memory (STM) [20].

1. Forget Gate: When LTM goes to the forget gate, it forgets unhelpful knowledge.
2. Learn Gate: Event (current input) and STM are integrated to enable the present input to be subjected to the recent knowledge we have gained via STM.
3. Remember Gate: LTM data that we haven't forgotten, together with STM and Event information, are integrated into Remember Gate to serve as an updated LTM.
4. Use Gate: To forecast the outcome of the current event, which serves as an updated STM, this gate additionally employs the LTM, STM, and Event.

The hidden layer cell state of the neuron, or the data held in the neuron, is an important component of the LSTM model. The cell states function on a data chain resembling the conveyor belt. There is just a limited amount of linear information interaction, which prevents information loss and successfully transfers information between long-short-term memory. The input gate, forget gate, update gate, and output gate make up an LSTM model [21].

4. K-fold cross-varidation

One statistical technique that may be used to gauge the effectiveness of machine learning models is cross-validation. Since it is simple to learn, simple to use, and produces skills estimate that it typically has a lower bias than other methodologies as the mean estimate of any parameter is less biased than a one-shot estimate, it is frequently used in involved machine learning to compare and select models for a given predictable modeling problem [22, 23]. There are four phases that make up the entire K-Fold cross-validation algorithm [24]:

1. Equally divide the training set S into k sections. Each subset has m/k training examples if the total number of training samples in S sets is m. The relevant subset is identified by the notation $\{S_1,S_2,....,S_k\}$.
2. Select one from the sample set to serve as m_i After that, choose k-1 from the training subset $\{S_1, S_2, S_{j-1}, S_{j+1},... S_k\}$. (There is just one person remaining who is S_j.) To obtain the hypothetical function h_{ij}, train with this k-1 subset m_i. Test the remaining S_j to see if there are any mistakes $\dot{\varepsilon}_{sj}(S_{ij})$.
3. We receive k empirical errors since we remove one S_j at once (j ranging from 1 to k). The average of these k empirical errors is what a $m_i$empirical error is as a result. The average of the values generated in the loop is the

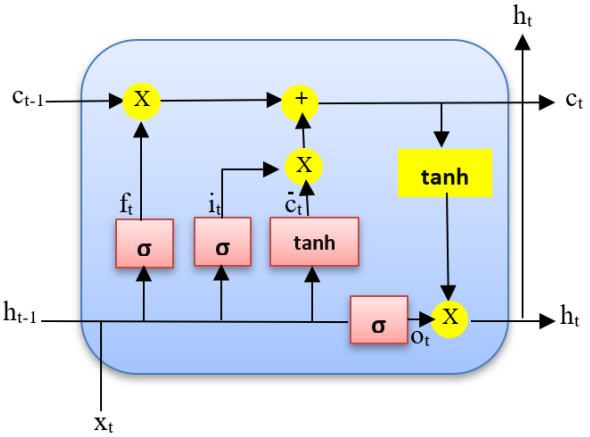


Figure. 1 Long short-term memory (LSTM) networks

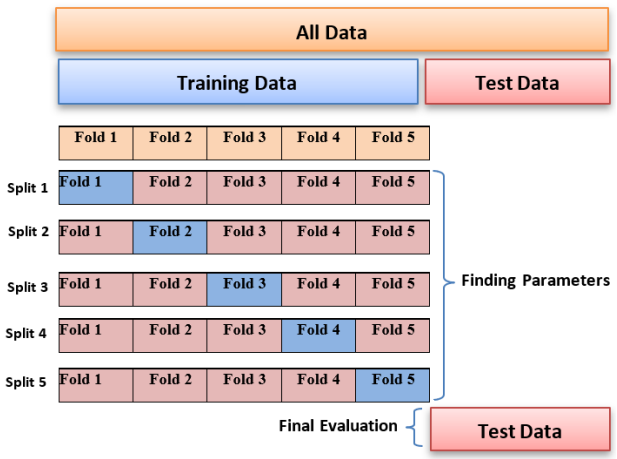


Figure. 2 5-fold cross-validation technique

performance indicator supplied by the K-Fold cross-validation.

4. Select the alternative with the smallest median empirical error rate m_i. Then perform another training to obtain the last h_i while using all S.

Although this approach might be computationally expensive, it does not waste a lot of data. That is a significant benefit when trying to solve issues with a small number of samples, like inverse inference [25]. Fig. 2 illustrates K-Fold Cross-Validation.

5. Methodology

In this work, we suggest a method for locating gene mutations in DNA and identifying their type. The method entails five steps:

1. Data collection.
2. Data pre-processing
3. Mutation detection
4. Data encoding
5. Mutation classification

5.1 Data collection

One of the mutation data sets is available at DepMap (https://depmap.org/portal/#). It is a Cancer Cell Lines portal (CCLE). It serves as a platform for the scientific community to learn about cancer vulnerabilities. The data set consists of a Mutation Annotation Formatfile (MAF) including information on all somatic point mutations and indels discovered in DepMap cell lines. The file contains a data set consisting of 1,048,575 rows representing different types of genetic mutations.

5.2 Data preprocessing

Data pre-processing transforms data into a format that machine learning can handle more quickly and effectively. In this step, the data set is prepared for the training and testing phase. Inappropriate and useless features of the dataset are eliminated. This stage includes the following steps:

A. Features Extraction: The CCLE dataset consists of 32 features that represent mutation metadata. The necessity to choose the essential features for the modified data set results from the fact that many characteristics are not required for training. The features chosen for the training process are Codon_Change and Variant_Classification.

B. Complex Features Simplification: The Codon_Change feature is a complex feature consisting of the mutation position, the reference codon, and the mutant codon. In this step, this feature is simplified by dividing it into three features to obtain new features that we need in the training to predict the type of mutation as shown in Table 1.

Table 1. Example of extract new features from the Codon_Change feature

CodonChange	Position	Reference codon	Mutant codon
c.(544-546)gaC>gaG	544-546	gaC	gaG
c.(6121-6123)tCa>tAa	6121-6123	tCa	tAa
c.(4666-4668)cGg>cAg	4666-4668	cGg	cAg

Table 2. Data layout

Position	Reference codon	Mutant codon
544-546	GAC	GAG
6121-6123	TCA	TAA
4666-4668	CGG	CAG

Table 3. Unsuitable data cleaning

Index	Reference codon
1	c.(937-969)agctcctctccccagccagaagaaaccac
2	c.(379-396)gggggcgggcccgcagfs
3	c.(244-288)cagaccactctggcgtctcctgaagtgaga

C. Data layout: The data is currently converted to produce codons in a format that only uses capital letters. as shown in Table 2.

D. Unsuitable Data Cleaning: The incorrect data is removed at this step, as seen in Table. 3.

E. Fill in the missing rows : In this step, the NULL rows that represent deleted or inserted codons are replaced by three gaps to represent the deleted/inserted codon (---).

5.3 Mutation detection

The first step in this research is finding mutations in order to forecast the types of mutations that will be found in the subsequent phase. At this point, the reference sequence and mutant sequence were compared using Needleman's mutation detection technique for global alignment. The Pseudo code of Needleman-Wunsch Algorithm used in the alignment process to detect the location of the mutation is shown in Algorithm 1.

In the following alignment, the reference sequences and mutant sequences are split into codons for initiation to the classification process as shown in Fig. 3.

Algorithm.1 Needleman-Wunsch Algorithm

M- sequence 1 length

N- sequence 2 length

S-M x Ndimensional dynamic programming matrix

$\delta(x_i,-)$-score from aligning x_i with a gap

$\delta(-,y_j)$-score from aligning y_j with a gap

$\delta(x_i,y_j)$-score from aligning xi with y_j

1. *S[0,0]=0*
2. *For i=1 to M Do:*
   - *S[I,0]=s[i-1,0]+ δ(xi,-)*
3. *For j=1 to N Do:*
   - *S[0,j]=S[0,j-1]+ $\delta(-,y_j)$*
   - *For i=1 to M Do:*

$$S[i,j]=MAX \begin{cases} S[i-1,j-1]+\delta(x_i,y_j) \\ S[i-1,j]+\delta(x_i,-) \\ S[i,j-1]+\delta(-,y_j) \end{cases}$$

4. *Return S[M,N]*

Output: *Optimal alignment score for aligning M and N*

Time Complexity: *O(MN)*

Reference Codons	Mutant Codons
CTA	GTA
GGG	– – –
GTC	GAC
C – C	CCC
AG–	– GT

Figure. 3 Splitting reference sequences and mutated sequences

5.4 Data encoding

A dictionary that translates the various symbols in the sequence (like ACGT) to a numeric code is used to turn the input into base sequences before feeding it into the deep neural network. Fig. 4 illustrates how concatenated characters might be mapped to tokens:

{'A': 1, 'C': 2, 'G': 3, 'T':4, '_':5}

Reference Codons	Mutant Codons
CTA	CTA
GGG	– – –
GTC	GTC
C – C	CCC
AAC	AAC
TCC	TCA
G – –	GCG
ATT	ATC
GTA	G – –

Reference Codons	Mutant Codons
241	241
333	555
342	342
252	222
112	112
422	421
355	323
144	142
341	355

Figure. 4 Data encoding

Table 4. Target encoding

	Mutation Type	One-Hot-Coding
1	Missense substitution mutation	100000000
2	Nonsense substitution mutation	010000000
3	Silent substitution Mutation	001000000
4	Read-through substitution mutation	000100000
5	Frameshift insertion mutation	000010000
6	Frameshift deletion mutation	000000100
7	in-frame insertion mutation	000000010
8	in-frame deletion mutation	000000001

While the target is encoded using one-hot-coding to represent all types of mutations represented by the four types of switch mutations (Missense mutation, Nonsense mutation, Silent Mutation, Read-through mutation) and insertion/deletion mutation (Frameshift mutation, in-frame mutation) as shown in Table 4.

5.5 Mutation classification

Mutations classification is the second stage of the suggested work. Building a classifier model is the first step in classifying DNA sequence alterations. These mutations are divided into three primary categories (deletions, insertions, and substitutions) as well as subcategories (frame-shift insertion, in-frame insertion, frameshift deletion, in-frame deletion, missense, silent, nonsense, and read-through).

The LSTM deep learning network model has been suggested to categorize different mutation types. When learning and hyperparameter tuning used 80% of the data set, while model testing occupied 20% of the data. In order to performthis, we conducta 10-fold validation by dividing training process into ten parts, so that the model is trained on nine parts and predicts the tenth part, and this is repeated ten times. The model was built using an LSTM technique, and several neural network architectures were tested to find the best one. The three layers of the network covered by this model are as follows: the embedding layer, the LSTM layer, and the dense layer. They used this cross-validation procedure to assess them, and the design with the greatest F1 score across all ten folds was chosen.

The proposed neural networks underwent 100 epochs of training, which means they had to go over the full dataset 100 times. Between each epoch, the model provided prediction on the validity fold, and it was recorded which epoch had the highest performance on the validation set. Figure 5 shows the classification process.

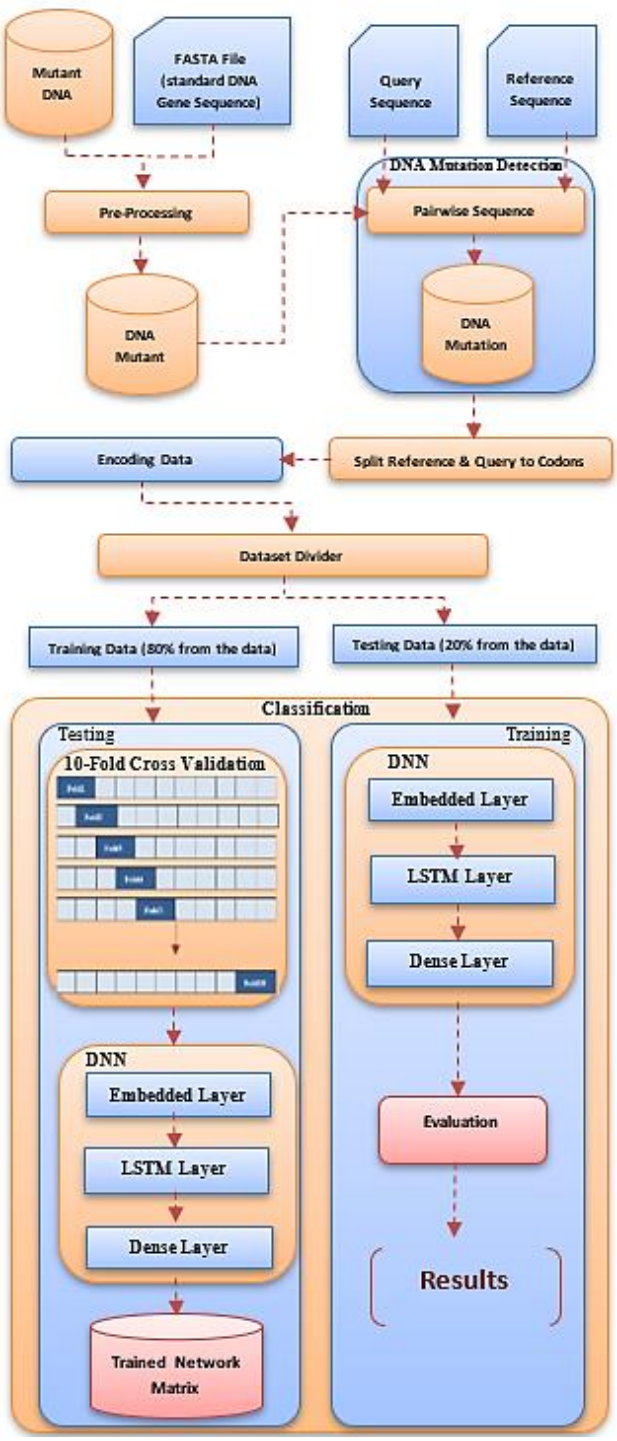


Figure. 5 Overall phases of the suggested mutation detection and classification method

The use of LSTM networks in this situation may be advantageous. A characteristic of LSTM allows data sequences to be preserved. It eliminates unused information, and as we all know, data is always filled with a lot of unused information that can be eliminated by LSTM in order to speed up computation and lower costs. For this reason, the ability of LSTM to eliminate unused information while maintaining information sequence makes it a potent tool for classification. A prediction model for DNA mutations was put out. With this strategy, we aimed to attain the best prediction accuracy by arranging the sequencing data in a unique manner. We suggested that the method be broken down into four key phases.

The sequence of data sets is preprocessed in the first stage. After the data sequence processing is finished in the second stage, the sequence is divided into codons (3 bases), and it is then turned into a format appropriate for training the LSTM network by and encoded {'A': 1, 'C': 2, 'G': 3, 'T':4, '_':5}. The target values are also encoded using one-hot encoding, where each class is represented by a binary vector with all '0' values except for the pointer to the word, which is set to 1. In the third stage, input data passed for training on the LSTM layer. In the last stage collected results are assessed.

In model Training, the reference codon and the query codon are given to the network as input. A hidden state vector and a cell state vector serve as the preceding cell's inputs for each LSTM cell. We output the hidden and cell-state vectors concatenated. The model's previous input for the subsequent generation, the hidden state, and the cell-state vectors from the previous cell are all inputs to each cell in the next step. The hidden state vector is fed to the dense layer (output layer), which outputs a posterior distribution over the next generation word at that place, in addition to passing those two vectors to the subsequent cell after updating them. A SoftMax function is used in order to obtain a probability distribution. Fig. 6 shows how network architecture works.

The quantity of information that may enter each node state is controlled by the LSTM using a number of gates. Below is a detailed explanation of the LSTM cell and its gates:

1) Forget gate:

$$f_t = \sigma(\omega_f \cdot [h_{t-1}, x_t] + b_f) \quad (1)$$

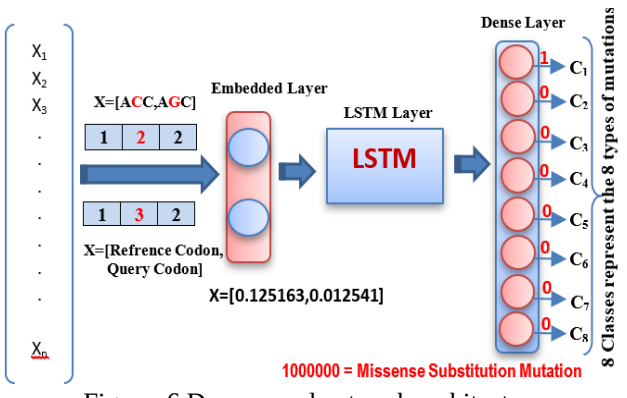


Figure. 6 Deep neural network architecture

2) Input gate:

$$i_t = \sigma(\omega_i \cdot [h_{t-1}, x_t] + b_i)$$
$$c_t = tanh(\omega_c \cdot [h_t-1, x_t] + b_c) \quad (2)$$

3) Cell state:

$$c_t = f_t \cdot c_{t-1} + i_t \cdot c_t \quad (3)$$

4) Output gate:

$$\sigma_t = \sigma(\omega_o \cdot [h_{t-1}, x_t] + b_o)$$
$$h_t = o_t \cdot tanh(c_t) \quad (4)$$

where x_t represents input on step time right now, σ denotes the logistics sigmoid function, and displays the elemental multiplication. Intuitively, the input gate regulates how much each unit is updated, the forget gate controls how many memory cells are removed per unit, and the gate outputs govern how much of the internal memory state is exposed.

6. Result

Due to the availability of large-scale biological data from high-throughput testing and the maturation of techniques for learning large numbers of network parameters, we can now evaluate the viability and utility of using specific deep neural network architectures as categorization tools for detecting genetic mutations. Numerous concerns arise as a result of the rapid uptake and rising popularity of deep learning materials to solve a wide range of biological issues, including the following: Can deep learning models successfully classify and differentiate between different mutation types using only nucleotide information? And what significance does the contribution of various structural and functional predictors derived from a biophysical approach have in this situation?

The CCLE dataset is utilized, and it includes a csv file with the features and the label, For the aim of identifying the mutation position, two fasta files representing the reference file (a file with no genetic mutations) and the query file (a file containing genetic mutations) were also utilized.The dataset consists of 32 features from which we derived features relevant to our study, and additional features from these features are extracted. The data set contains 1005217mutations of various types, as shown in Table 5 and the distribution of mutation types in the dataset is shown in Fig. 7.

The alignment that this analysis produced is as follows:

ACTG-ATTCA……n+1 (5)

AC-GCAT-CA……m+1 (6)

Table 5. Mutations type

	Mutation Type	Number of Mutation
1	Silent substitution mutation	305193
2	Missense substitution mutation	594942
3	Nonsense substitution Mutation	34659
4	Read-through substitution mutation	832
5	Frameshift deletion mutation	37205
6	in-frame deletion mutation	6481
7	Frameshift insertion mutation	23182
8	in-frame insertion mutation	2723

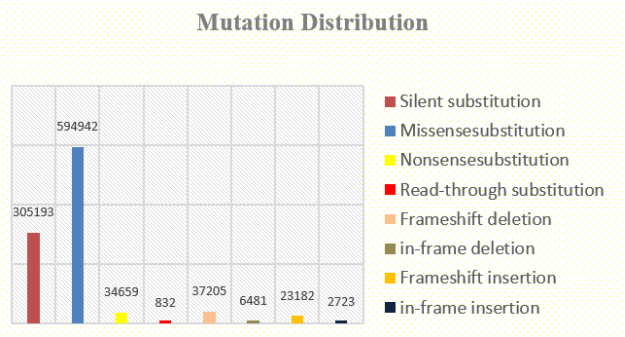


Figure. 7 Mutation types distribution

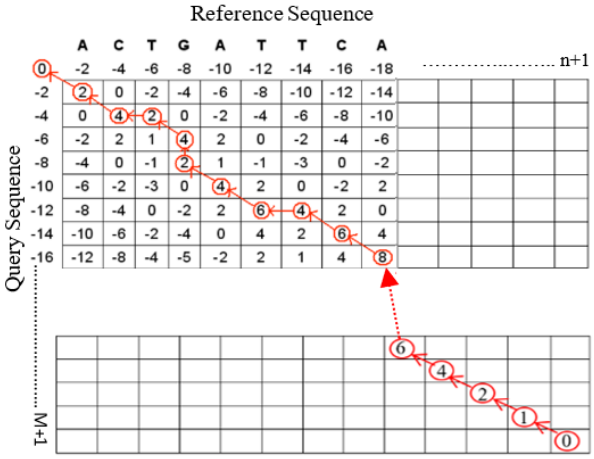


Figure. 8 Sequence alignment using needleman algorithm

The first stage of the work consists of using Needleman algorithm to align the reference and mutant sequences, by comparing the two sequences, in order to identify the positions of mutations as shown in Fig. 8.

In the following alignment, the reference and query sequences are divided into codons (3 bases), which are then ready to enter into the deep neural network for training and testing. After the data was divided into codons, the neural network training procedure started. The data are divided into two parts: 80% of it was used for training and 20% of it was used for testing.To anticipate and classify mutations, we investigated Four machine learning classifiers,

namely Decision Tree, Logistic Regression, KNN, and Naive Bayes. They were applied to the data for the purpose of comparing them with the proposed model. We trained the network on mutations within the CCLE database using these classifiers and these classifiers were evaluated using the following scales:

1. Accuracy: It measures the proportion of accurate predictions to all input samples.

$$\text{Accuracy} = \frac{\text{Number of correct preductions}}{\text{Total number of preductions made}} \quad (7)$$

2. Precision: It is calculated as the proportion of co rrect positive findings to those that the classifier anticipated to be positive.

$$\text{Precision} = \frac{\text{True positives}}{\text{True positives+False positives}} \quad (8)$$

3. Recall: It is calculated as the total number of rele vant samples divided by the number of correct p ositive outcomes

$$\text{Recall} = \frac{\text{True positives}}{\text{True positives+False negative}} \quad (9)$$

4. F1-Score:The Harmonic Mean of recall and prec ision is the F1 Score. F1 Score has a range of [0, 1]. It reveals the precision and durability of you r classifier (the proportion of properly classified cases)

$$\text{F1} - \text{Score} = 2 * (\frac{\text{Precision} * \text{Recall}}{\text{Precision+ Recall}}) \quad (10)$$

Table 6 shows a comparison between the classifi ers that were tested on our data set according to the a bove-mentioned metrics.

We trained various deep neural networks to reach the highest possible accuracy for predicting the types of genetic mutations, where LSTM was used, and the accuracy was 99.97%. It gives the expected accuracy, then K-Fold cross-validation was used which in turn divides the training data into 10 folds so that the network can train and give better accuracy, and this has achieved an accuracy of 100. By testing and classifying our dataset using several classifiers such as (Naive Bayes, Logistic Regression, Decision Tree, KNN, CNN+LSTM) and comparing it with the proposed model, we found that the accuracy gained from the proposed model is higher than the accuracy that we obtained from other classifiers, as shown in Table 7.

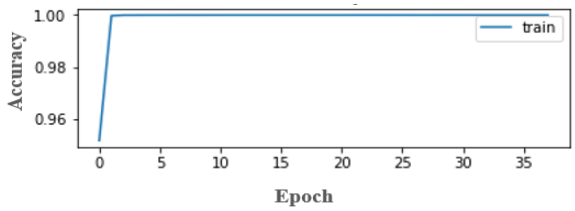
Table 6. Comparison between different classifiers

Strategy	Accuracy	precision	recall	F1-score
Naive Bayes	57.88%	48%	49%	44%
LogisticRegression	59.60%	20%	16%	14%
DecisionTree	60.95%	33%	32%	31%
KNN	96.15%	90%	80%	87%

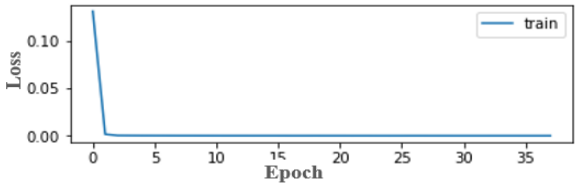
Table 7. Comparison between the suggested approach with alternative strategies

Strategy	Accuracy	precision	recall	F1-score
Naive Bayes	66.66%	23%	33%	27%
LogisticRegression	66.91%	30%	16%	16%
Decision Tree	74.01%	18%	15%	15%
CNN+LSTM	84.30%	82%	37%	45%
KNN	98.84%	90%	81%	89%
LSTM	99.97%	100%	97%	98%
LSTM+K-Fold cross-validation	**100%**	**100%**	**100%**	**100%**

When training data is conveyed via a network, its primary goal is to minimize loss, either in terms of errors or costs, that is seen in the output. After calculating the gradient, or loss in relation to a particular set of weights, the weights are modified appropriately. We continue using these weights until we discover the ideal weights, for which the loss is the least. A critical step in evaluating the effectiveness of any model in deep learning is to check the accuracy of the model after the training phase. Fig. 9 shows the verification of the increase in accuracy in learning. Fig. 9(a) shows the progress in learning and the increase in accuracy from the initial stages of training. Fig. 9(b) shows the decrease in the amount of loss from the initial stages of training as well.



(a)Accuracy



(b)Loss

Figure. 9 Accuracy and loss result from the proposed model

7. Comparative analysis

The proposed technique has also been compared with the methodologies that were reviewed in previous studies included in Section 2 of this research. The comparison can be summarized as follows:

When comparing the proposed model with the model presented in reference [9], it is found that the Scaling Petri nets is tricky. As a result, attempts to reproduce biological systems using standard Petri nets have mostly focused on extremely tiny models up until now. In contrast, the proposed system showed its ability to be used in large models.

The researchers at reference [10] used logistic regression. Complex relationships are challenging to get using logistic regression. This approach can be readily defeated by the deep neural network techniques utilized in our suggested technique.

The model used in reference [11] showed that it is able to predict with much less accuracy than that achieved in our proposed technique.

The SVM was used as one of the machine learning techniques in the reference [12]. Due to the fact that this algorithm is not suitable for very large datasets, the technique proposed in this paper is considered more efficient for classifying large datasets. Because the length of the DNA sequence can reach millions of nitrogenous bases, and this requires a technique that works very efficiently with very large sequence lengths.

The researchers of the reference [13] published a study that included methods for categorizing comparable mutants using only two types of mutation operators, namely ABS (Absolute Value Insertion) and UOI (Unary Operator Insertion). But the proposed technique in this paper can detect and determine the types of all forms of mutations.

Table 8. Comparison between the suggested approach with related works

Reference No.	Technique	Accuracy
[9]	Petri Nets	-
[10]	logistic regression	64%
[11]	CNN and BiGRU Network	84%
[12]	SVM and one hot coding	91%
[13]	Abstract Syntax Tree Neural Networks	90%
[14]	Natural Language Processing	70%
[15]	Sequential Labeling Model	95%
Propose Technique	**LSTM+K-Fold cross-validation**	**100%**

The researchers of reference [14] have used Natural Language Processing (NLP) to forecast the sorts of genetic changes, however finding effective NLP techniques is challenging, making it challenging to finish this task.

In contrast to our technique, which identifies the location of the mutation, its main types, and mutation subtype, the model in the reference [14] classifies the three primary types of mutations (deletion, substitution, and insertion) only. Also, the accuracy of the model is lower than the accuracy of proposed technique. The accuracy of the methods in the aforementioned studies and the one that is suggested in this paper are presented in Table 8.

8. Conclusion

It is essential to classify genetic mutations in order to understand the nature and position of the mutation and to quickly recognize the disease that the mutation causes. As a result, the goal of this study was to propose a deep neural network model based on the LSTM network. To do this, we extracted the features from the data set and trained the network using 80% of the Cancer Cell Lines portal (CCLE) data set by splitting the training data into 10 folds. By comparing our results with those of other strategies, we discovered that the proposed approach is capable of detecting and classifying the primary types of genetic mutations (deletions, insertions, and substitutions) and subtypes of genetic mutations represented by (frame-shift insertion, in-frame insertion, frameshift deletion, in-frame deletion, missense, silent, nonsense, and read-through) with greater accuracy than other strategies, with an accuracy of 100%.The quantity of training data and training epoch utilized have an impact on the classification accuracy. During training, it's crucial to establish the right parameters, including the ideal function, learning rate, meta-architecture, and input sample size, in order to achieve the greatest accuracy. The quality of the dataset and accurate data annotation are crucialto achieving the highest possible accuracy. Also,the form of LSTM itself exhibits temporal behaviour and sequentially captivates the data, which is a more suitable technique in sequences classification, the experiment found that LSTM is acceptable with genetic mutation classification issues.

Conflicts of Interest

The authors declare no conflict of interest.

Author Contributions

Rana H. Saloom provided the idea, technique, software, formal analysis, materials, data collection, and writing-original version preparation. Hussein K. Khafaji provided supervision, revision, and editing.

References

[1] A. Khan and B. Lee, "Gene transformer: Transformers for the gene expression-based classification of lung cancer subtypes", *arXiv Preprint arXiv:2108.11833*, 2021.

[2] V. F. Gonçalves, "Mitochondrial genetics", *Mitochondria in Health and in Sickness*, pp. 247-255, 2019.

[3] C. Lawless, L. Greaves, A. K. Reeve, D. M. Turnbull, and A. E. Vincent, "The rise and rise of mitochondrial DNA mutations", *Open Biol*, Vol. 10, No. 5, p. 200061, 2020.

[4] Y. Luo, J. Ma, and W. Lu, "The significance of mitochondrial dysfunction in cancer", *Int J Mol Sci*, Vol. 21, No. 16, p. 5598, 2020.

[5] A. Natarajan and T. Ravi, "A survey on gene feature selection using microarray data for cancer classification", *International Journal of Computer Science & Communication (IJCSC)*, Vol. 5, No. 1, pp. 126-129, 2014.

[6] L. Rylaarsdam and A. G. Gamboa, "Genetic causes and modifiers of autism spectrum disorder", *Front Cell Neurosci*, p. 385, 2019.

[7] A. E. E. D. Rashed, H. M. Amer, M. E. Seddek, and H. E. D. Moustafa, "Sequence Alignment Using Machine Learning-Based Needleman-Wunsch Algorithm", *IEEE Access*, Vol. 9, pp. 109522-109535, 2021.

[8] X. Xu, Y. Chan, K. Xu, J. Zhang, X. Wang, Z. Yin, and W. Liu, "SLPal: Accelerating long sequence alignment on many-core and multi-core architectures", In: *Proc. of 2020 IEEE International Conference on Bioinformatics and Biomedicine (BIBM)*, pp. 2242-2249, 2020.

[9] J. Yang, J. Lian, H. Pu, and J. Gu, "Modeling and analysis of DNA mutation type based on colored Petri net", In: *Proc. of 2017 IEEE International Conference on Information and Automation (ICIA)*, pp. 864-869, 2017.

[10] R. N. Waykole and A. D. Thakare, "Intelligent classification of clinically actionable genetic mutations based on clinical evidences", In: *Proc. of 2018 Fourth International Conference on Computing Communication Control and Automation (ICCUBEA)*, pp. 1-4, 2018.

[11] J. Xu, X. Zheng, and M. Jiang, "Gene mutation classification using CNN and BiGRU network", In: *Proc. of 2019 9th International Conference on Information Science and Technology (ICIST)*, pp. 397-401, 2019.

[12] V. K. Singh, N. S. Maurya, A. Mani, and R. S. Yadav, "Machine learning method using position-specific mutation based classification outperforms one hot coding for disease severity prediction in haemophilia 'A'", *Genomics*, Vol. 112, No. 6, pp. 5122-5128, 2020.

[13] S. Peacock, L. Deng, J. Dehlinger, and S. Chakraborty, "Automatic equivalent mutants classification using abstract syntax tree neural networks", In: *Proc. of 2021 IEEE International Conference on Software Testing, Verification and Validation Workshops (ICSTW)*, pp. 13-18, 2021.

[14] M. Gupta, H. Wu, S. Arora, A. Gupta, G. Chaudhary, and Q. Hua, "Gene Mutation Classification through Text Evidence Facilitating Cancer Tumour Detection", *J HealthcEng*, Vol. 2021, 2021.

[15] U. N. Wisesty, A. Purwarianti, A. Pancoro, A. Chattopadhyay, N. N. Phan, E. Y. Chuang, and T. R. Mengko, "Join Classifier of Type and Index Mutation on Lung Cancer DNA Using Sequential Labeling Model", *IEEE Access*, Vol. 10, pp. 9004-9021, 2022.

[16] K. Ma and H. Leung, "A novel LSTM approach for asynchronous multivariate time series prediction", In: *Proc. of 2019 International Joint Conference on Neural Networks (IJCNN)*, pp. 1-7, 2019.

[17] J. Ostmeyer and L. Cowell, "Machine learning on sequential data using a recurrent weighted average", *Neurocomputing*, Vol. 331, pp. 281-288, 2019.

[18] B. N. Saha and A. Senapati, "Long Short Term Memory (LSTM) based Deep Learning for Sentiment Analysis of English and Spanish Data", In: *Proc. of 2020 International Conference on Computational Performance Evaluation (ComPE)*, pp. 442-446, 2020.

[19] N. S. Malinović, B. B. Predić, and M. Roganović, "Multilayer long short-term memory (LSTM) neural networks in time series analysis", In: *Proc. of 2020 55th International Scientific Conference on Information, Communication and Energy Systems and Technologies (ICEST)*, pp. 11-14, 2020.

[20] Y. Lu and F. M. Salem, "Simplified gating in long short-term memory (lstm) recurrent neural networks", In: *Proc. of 2017 IEEE 60th International Midwest Symposium on Circuits and Systems (MWSCAS)*, pp. 1601-1604, 2017.

[21] T. Liu, T. Wu, M. Wang, M. Fu, J. Kang, and H. Zhang, "Recurrent neural networks based on

LSTM for predicting geomagnetic field", In: *Proc. of 2018 IEEE International Conference on Aerospace Electronics and Remote Sensing Technology (ICARES)*, pp. 1-5, 2018.

[22] J. Pohjankukka, T. Pahikkala, P. Nevalainen, and J. Heikkonen, "Estimating the prediction performance of spatial models via spatial k-fold cross validation", *International Journal of Geographical Information Science*, Vol. 31, No. 10, pp. 2001-2019, 2017.

[23] K. Pal and B. V. Patel, "Data classification with k-fold cross validation and holdout accuracy estimation methods with 5 different machine learning techniques", In: *Proc. of 2020 Fourth International Conference on Computing Methodologies and Communication (ICCMC)*, pp. 83-87, 2020.

[24] Z. Xiong, Y. Cui, Z. Liu, Y. Zhao, M. Hu, and J. Hu, "Evaluating explorative prediction power of machine learning algorithms for materials discovery using k-fold forward cross-validation", *Comput Mater Sci*, Vol. 171, p. 109203, 2020.

[25] Y. Nie, L. D. Santis, M. Carratù, M. O'Nils, P. Sommella, and J. Lundgren, "Deep melanoma classification with K-fold cross-validation for process optimization", In: *Proc. of 2020 IEEE International Symposium on Medical Measurements and Applications (MeMeA)*, pp. 1-6, 2020.