DNA sequence classification by Deep Neural Network

Md. Tarek Hasan, Mohammed Jawwadul Islam, Arifa Akter, Sumayra Islam, Mohammad Fahad Al Rafi
Department of Computer Science and Engineering, United International University
Plot-2, United City, Madani Avenue, Badda, Dhaka-1212, Bangladesh
(mhasan181076, mislam181182, aakter181254, sislam181123, mrafi181201)@bscse.uiu.ac.bd

Introduction

Motivation

The study of DNA is a vital part of understanding organisms in life science. In all species, DNA contains the majority of the genetic instructions for development, function, and reproduction. In the field of bioinformatics, the classification of DNA sequences is a very important task. Through classification, we can survey and track various organisms and their evolutionary traits. So, if we can correctly classify DNA sequences, we will be able to survey and track diverse organisms and their evolutionary features. DNA sequences are very long, for example, small bacterial genomes are a few million base pairs. So, to classify these sequences would require powerful algorithms capable of performing lots of computation.

Histones are highly basic proteins with a lot of lysine and arginine residues that are present in the nucleus of eukaryotic cells. Histones protect DNA from tangles and damage by preventing it from getting knotted. Histones also play a crucial function in gene regulation and DNA replication. Unwound DNA in chromosomes would be very lengthy if histones were not there. By classifying the histone protein in the DNA sequence, we will be able to get a clear idea about the histone that is responsible for certain changes or vice-versa which has motivated us to work with the DNA sequence classification using the dataset.

Deep learning is one of the most extensively utilized approaches for any type of classification task since it automates the process of extracting features from data. In recent years, deep learning has outperformed traditional machine learning algorithms in most of the classification tasks in the area of Bioinformatics. That is why we have chosen a classification task to do an experimental analysis on different deep learning algorithms and sequence embedding techniques as the main challenge for the DNA classification task is feature extraction from the sequences.

History of the Problem

Over the last few years, scientists have been collecting a lot of information on DNA sequences. As a result, the amount of data on DNA sequences has been increasing exponentially every year. To work with these sequences, to understand them, researchers have been using the power of modern computation provided due to advances in technology, particularly accessibility of

superior computing hardware (e.g. GPUs) required to handle DNA sequences. Living organisms are classified with specific scientific names. This classification is done on a hierarchical basis, which allows researchers to keep track of parent-child organisms.

Throughout the years, machine learning models were trained on DNA sequences to predict the class of unknown organisms. However, in recent years, a new technique called deep learning was introduced. Deep learning is a branch of machine learning where artificial neural networks are used. These are algorithms inspired by neurons inside the human brain. With many nonlinear layers, each layer representing data at high-level abstractions, deep learning algorithms solve complex problems by training the neural networks on large amounts of data. Therefore, several researchers have started applying deep learning models for studying and understanding DNA sequences.

Problem Description

Deep learning models have been used to extract features of high-level abstraction. One such model is called Convolutional Neural Network (CNN) [19]. In a convolutional neural network model, neurons in a convolutional layer can extract higher-level abstraction features from extracted features from the previous layer. Another model is called Bidirectional Long Short-Term Memory (Bi-LSTM) [20]. Bi-LSTM is just two separate RNNs combined. This structure enables the networks to contain both backward and forward sequence information at each time step.

In this work, we have worked on the DNA sequence classification problem where the input is the DNA sequence and the output class states whether a certain histone protein is present on the sequence or not. For this purpose, we have used one of the datasets from 12 different datasets [1] that we have collected. The name of the dataset is H3K4me2. Because of the time limitation, just one dataset was chosen, and H3K4me2 was chosen because prior research on this dataset had shown unsatisfactory results. The main challenge of our work was feature extraction from the DNA sequences. To represent a sequence, we have utilized k-mer representation. For the sequence embedding we have used one-hot encoding, and two different word embedding models, one is Word2Vec [16,17] and another one is BERT [18]. Moreover, we have used Keras Embedding layer, Bi-LSTM, and CNN for our experiments.

Literature Review

In the domain of Bioinformatics, there are several works available for the sequence classification task. We are presenting some of the most related works compared to our work. For a better understanding of this part, we are using a tabular format to present the previous studies. To give a comprehensive view of that task, Table 1 shows the major work that has been done by other researchers, dataset, feature extraction techniques, and deep learning models used by them.

Reference	Major Work	Dataset(s)	Feature Extraction Technique	Deep learning/ Machine Learning Model
Nguyen et al. [1]	Classifying DNA sequences	H3, H4, H3K9ac, H3K14ac, H4ac, H3K4me1, H3K4me2, H3K4me3, H3K36me3, H3K79me3, Splice, Promoter	One-hot encoding	CNN
Yin et al. [2]	An image-representation of primary DNA sequence as its input, and predicts key determinants of chromatin structure	Doulrs/Hilbert-CNN: An image representation based convolutional network for DNA classification (github.com)	K-mer frequency, One hot label encoding	CNN, Kernel SVM, LSTM
Lopez-Rincon et al. [3]	an assisted detection test, combining molecular testing with deep learning. Automatically create features starting from the genome sequence of the virus.	albertotonda/deep-learning-coronavirus-genome: Repository with data and code for the paper "Identification of SARS-CoV-2 from Genome Sequences using Deep Learning" (github.com)	PCR Amplification, One hot Encoding	CNN
Phan et al. [4]	an effective framework for improving fixed-length DNA	btu083 Supplementary Data	K-mer frequency	Kernel SVM, Random Forest Classifier

	sequence classification by using the combination of categorical features and numerical features			
Lopez-Rincon et al. [5]	discovery of representative genomic sequences in SARS-CoV-2	steppenwolf0/primers-sars-cov-2 (github.com)	One-hot label encoding	CNN
Kassim et al. [6]	DNA Sequence Classification (HBV, non-HVB nucleotide sequences)	Leung et al. [7]	Sequence of Words	CNN
Whata et al. [8]	DNA Sequence Classification (SARS COV-2 Genome	SARS-CoV2_20210514	CNN, max-pooling	CNN, CNN, Bi-LSTM
Zhang et al. [9]	DNA Sequence Classification (SARS COV 2)	https://ftp.ncbi.nlm.nih.gov/	CNN	CNN, CNN LSTM, CNN, Bi-LSTM
Bosco et al. [10]	DNA Sequence Classification	16S dataset	character-level-one-hot-encoding	CNN LSTM
Rizzo et al. [11]	DNA Sequence Classification	16S rRNA	k-mer representation	CNN
Gunasekaran et al. [12]	DNA Sequence Classification	The complete DNA/Genomic sequence of the viruses like COVID, SARS, MERS, dengue, hepatitis, and influenza obtained from the public nucleotide sequence	Label and K-mer encoding	CNN, CNN-LSTM, CNN-Bidirectional LSTM

		database: "The National Centre for Biotechnology Information (NCBI)" (https://www.ncbi.nlm.nih.gov).		
Akkaya et al. [13]	DNA Sequence Classification	splice, promoter, and H3	one-hot based with random and default dictionary, Voss and dna2vec	Deep Convolutional Neural Network
Mahmoud et al. [14]	DNA Sequence Classification	H3, H4, H3K9ac, H3K14ac, H4ac	DDV software to read the sequences and convert it into images	Xception network, VGG network, Pseudoinverse Learning Autoencoder (PILAE)
Higashihara et al. [15]	Classifying DNA sequences	H3, H4, H3K9ac, H3K14ac, H4ac, H3K4me1, H3K4me2, H3K4me3, H3K36me3, H3K79me3	4-mer frequency	SVM with RVF kernel

Table 1: Related studies on DNA sequence classification using Deep Learning Algorithms

We can get some important insights from the related works those have been done by other researchers. CNN is widely used for the sequence classification task along with the K-mer frequency representation, and one-hot encoding technique for sequence representation, and embedding. In addition, LSTM and Bi-LSTM are two other common deep learning models for sequence classification tasks. Some of the related works employed traditional machine learning algorithms, i.e. Support Vector Machine (SVM) with RBF kernel, Random Forest Classifier which is an ensemble learning technique for their experimental analysis.

Materials and Methods

Dataset Description

DNA sequences wrapped around histone proteins are the subject of datasets.

DNA's spatial organization is done by integrating (or "recruiting") additional molecules, such as histone proteins, which assist in assuming the proper spatial arrangement. Chromatin is the combination of DNA and helper molecules. To estimate chromatin state from DNA sequence characteristics, the datasets below can be used.

We have collected the dataset from Nguyen who is one of the authors of the paper titled “DNA sequence classification by convolutional neural network” [1]. The description of the dataset is explained in Table 2. For our experiment, we selected one of the datasets entitled H3K4me2. H3K4me2 has 30683 DNA sequences whose 18143 samples fall under the positive class, the rest of the samples fall under the negative class, and it makes the problem binary class classification. The ratio of the positive-negative class is around (59:41)%. The class label represents the presence of H3K4me2 histone proteins in the sequences. The base length of the sequences is 500. Moreover, we will employ other datasets for future experiments as an expanded version of our work.

No	Dataset	Description	Number of Classes	Number of Samples	Sequence Length (base)
1	H3	H3 occupancy	2	7667; 7298	500
2	H4	H4 occupancy	2	6480; 8121	500
3	H3K9ac	H3K9 acetylation relative to H3	2	15,415; 12,367	500
4	H3K14ac	H3K14 acetylation relative to H3	2	18,771; 14,277	500
5	H4ac	H4 acetylation relative to H4	2	18,410; 15,686	500
6	H3K4me1	H3K4 mono-methylation relative to H3	2	17,266; 14,411	500
7	H3K4me2	H3K4 di-methylation relative to H3	2	18,143; 12,540	500
8	H3K4me3	H3K4 tri-methylation relative to H3	2	19,604; 17,195	500
9	H3K36me3	H3K36 tri-methylation relative to H3	2	18,892; 15,988	500
10	H3K79me3	H3K79 tri-methylation relative to H3	2	15,337; 13,500	500

11	Splice	Primate splice-junction gene sequences with associated imperfect domain theory	3	762; 765; 1648	60
12	Promoter	*E. coli* promoter gene sequences with partial domain theory	2	53;53	57

Table 2: Description of the data set

Methodology

Data Cleaning

The datasets were gathered in.txt format. We discovered that the dataset contains id, sequence, and class label during the Exploratory Data Analysis phase of our work. We dropped the id column from the dataset because it is the only trait that all of the samples share. Except for two samples, H3K4me2 includes 36799 DNA sequences, the majority of which are 500 bases long. Those two sequences have lengths of 310 and 290, respectively. To begin, we employed the zero-padding strategy to tackle the problem. However, because there are only two examples of varying lengths, we dropped those two samples from the dataset later for experiments, as these samples may cause noise.

Sequence Representation

DNA sequence representation techniques are utilized to effectively designate a gene structure and facilitate coding sequence similarity/dissimilarity analysis. Numerical, Graphical, Geometrical, and Hybrid representation approaches are the most common. In our work, we have used the K-mer sequence representation technique. More specifically, we have used the 3-mer representation technique.

Example:

DNA Sequence before applying the 3-mer representation technique:

CATGTAGTATTGGGC

The Sequence after applying the 3-mer representation technique:

CAT ATG TGT GTA TAG AGT GTA TAT ATT TTG TGG GGG GGC

Sequence Embedding

The main challenge for DNA sequence classification task is the sequence embedding. It’s difficult due to the fact that sequence data is inherently unstructured. Sequences are arbitrary strings, just like texts in

Natural Language Processing (NLP). These strings have no significance for a computer. That's why we need to convert the strings into the numerical format.

For embedding the sequence after applying the 3-mer representation technique, we have experimented using six different embedding techniques. The first three embedding methods are named *SequenceEmbedding1D*, *SequenceEmbedding2D*, *SequenceEmbedding2D_V2*. The other two methods are Word2Vec [16,17] and BERT [18]. The base method of the first three methods is one-hot encoding. One-hot encoding represents a single 3-mer using a vector of size 64 of values 0 and 1, where there is only one 1 and the rest of the values are 0. There are 64 different 3-mer possible from the 4 nucleotides (*A*, *C*, *G*, *T*). To represent the one-hot encoding technique sets,

AAA: the first value of the vector as 1 and the rest of the values are 0,

AAC: the second value of the vector as 1 and the rest of the values are 0,

AAG: the third value of the vector as 1 and the rest of the values are 0, and so on.

SequenceEmbedding1D is the one-dimensional representation of a single DNA sequence which is basically the one-hot encoding. *SequenceEmbedding2D* is the two-dimensional representation of a single DNA sequence where the first row is the one-hot encoding of a sequence after applying 3-mer representation. The second row is the one-hot encoding of a left-rotated sequence after applying 3-mer representation. Similarly, the third row of *SequenceEmbedding2D_V2* is the one-hot encoding of a right-rotated sequence after applying 3-mer representation.

Word2Vec and BERT are the word embedding techniques for language modeling. But, Word2Vec is a widely used technique in the domain of Bioinformatics for sequence classification tasks as well. Moreover, we used one_hot from Keras.preprocessing.text which converts each 3-mer using an integer in the range from 1 to 64 uniquely.

Deep Learning Models

After the completion of sequence embedding, we have used deep learning models for the classification task. We have used two different deep learning models for this purpose, one is Convolutional Neural Network (CNN) [19] and the other is Bidirectional Long Short-Term Memory (Bi-LSTM) [20]. The details of the architectures are described in the following paragraphs.

Two one-dimensional convolutional layers with filter size 16 and kernel size 4 make up the CNN model. A one-dimensional max-pooling layer follows each of these layers. These layers are responsible for extracting features from sequence representation matrices. A fully connected neural network layer with 100 neurons is then used to convert the retrieved features. To mitigate the effect of overfitting, we employed a dropout [22] value of 0.5 in this layer. After that, input sequence labels are predicted using a softmax output layer. For all other layers, we have used the activation function: ReLU [23]. In Table 3, you can see the tabular format of the architecture. The hyperparameters that are not mentioned in Table 3 have been used as the default value.

Table 3: CNN Architecture for our experiments

Layer	Hyperparameters

1D Convolution	filter size = 16, kernel size = 4, activation function = ReLU
1D Max-pooling	pool size = 2
1D Convolution	filter size = 16, kernel size = 4, activation function = ReLU
1D Max-Pooling	pool size = 2
Flatten	
Dense	units = 100, activation function = ReLU
Dropout	rate = 0.5
Dense	units = 2, activation function = softmax

In the implementation of Bi-LSTM architecture, there is an embedding layer of Keras with input dimension 64, output dimension 256 and the input length is 498 as we can take 498 3-mer from a sequence of length 500. After this layer, we utilized a Spatial Dropout [21] Layer with a rate of 0.2. Then, a Bi-LSTM has been employed with 256 units. A fully connected neural network layer with 128 neurons is then used to convert the retrieved features. To prevent the effect of overfitting, we have used a dropout [22] with a ratio of 0.5 in this layer. Then, input sequence labels are predicted using a softmax output layer. For all other layers, we have used the activation function: ReLU [23]. In Table 4, you can see the tabular format of the architecture. The hyperparameters that are not mentioned in Table 4 have been used as the default value.

Table 4: Bi-LSTM Architecture for our experiments

Layer	Hyperparameters
Embedding	input dimension = 64, output dimension = 256, input length = 498
1D Spatial Dropout	rate = 0.2
Bidirectional LSTM	units = 256, return sequences = True
Flatten	
Dense	units = 128, activation function = ReLU
Dropout	rate = 0.5
Dense	units = 2, activation function = softmax

Finally, for both models, we have utilized adam as the optimizer, sparse categorical cross-entropy as the loss function. In addition, we have used early stopping by monitoring the validation accuracy with

patience for 200 epochs, and model checkpoint to save the weight of the best model. We have run the model for 300 epochs utilizing the model checkpoint and early stopping [24] with batch size 256.

For a better understanding of the implementation, we are adding the links to the supplementary materials:

CNN & Sequence Embedding Implementation, Bi-LSTM Implementation.

Experimental Analysis

All the experiments were conducted on 3.60GHz Intel(R) Core(TM) i7-7700 CPU, 32 GB RAM, and an NVIDIA TITAN XP with 12GB physical memory under Ubuntu 16.04.7 LIS. After the data cleaning phase, we had 36797 samples. We have used 80% of the whole dataset for training and the rest of the samples for testing. The dataset has been split using train_test_split from sklearn.model_selection stratifying by the class label. We have utilized 10% of the training data for validation purposes. For the first five experiments shown in Table 5, we have used batch training as it was throwing an exception of resource exhaustion.

The evaluation metrics we used for our experiments are accuracy, precision, recall, f1-score, and Matthews Correlation Coefficient (MCC) score. The minimum value of accuracy, precision, recall, f1-score can be 0 and the maximum value can be 1. The minimum value of the MCC score can be -1 and the maximum value can be 1.

The experimental results are shown in Table 5 where bold font represents the best score for the certain evaluation matrix.

Table 5: Result of the Experiments

Sequence Embedding Technique	Deep Learning Model	Accuracy	Precision	Recall	F1-Score	MCC Score
SequenceEmbedding1D	CNN	**0.6027**	**0.6467**	0.7230	0.6827	**0.1573**
SequenceEmbedding2D	CNN	0.5914	0.5914	1	**0.7432**	0
SequenceEmbedding2D_V2	CNN	0.5914	0.5914	1	**0.7432**	0
Word2Vec	CNN	0.5914	0.5914	1	**0.7432**	0
BERT	CNN	0.5914	0.5914	1	**0.7432**	0
one_hot	Bi-LSTM	0.5914	0.5914	1	**0.7432**	0

From the results of these experiments, we can state that we have got the maximum accuracy of 60.27%, precision 64.67%, and MCC score 0.1573 using *SequenceEmbedding1D* and CNN model. We have got the maximum recall score and f1-score on the rest of the experiments.

Discussion

MCC score 0 indicates the model's randomized predictions. The recall score indicates how well the classifier can find all positive samples. We can say that the model's ability to classify all positive samples has been at an all-time high over the last five experiments. The highest MCC score we received was 0.1573, indicating that the model is very near to predicting in a randomized approach. We attain a maximum accuracy of 60.27%, which is much lower than the state-of-the-art result of 71.77% [1]. To improve the score, we need to emphasize more on the sequence embedding approach. Furthermore, we can experiment with various deep learning techniques.

Conclusion

We used a variety of sequence embedding methods and deep learning models in our work. We can gain a thorough understanding of the combinations of sequence embedding techniques and deep learning models through this study. We can conclude that the aforementioned combinations are ineffective for DNA sequence classification tasks. We will test other combinations of the above-mentioned embedding approach and DL models, i.e. Word2Vec with Bi-LSTM, as an expanded version of our work. We will also test different deep learning models on other datasets, such as Bi-LSTM with an attention mechanism and GRU [25]. In addition, we can tune the K value for the K-mer representation.

Acknowledgment

We gratefully acknowledge the support of NVIDIA Corporation with the donation of the Titan Xp GPU and Nguyen to provide us with the datasets used for this work.

References

[1] Nguyen, N.G., Tran, V.A., Phan, D., Lumbanraja, F.R., Faisal, M.R., Abapihi, B., Kubo, M. and Satou, K., 2016. DNA sequence classification by convolutional neural network. Journal Biomedical Science and Engineering, 9(5), pp.280-286.

[2] Yin, B., Balvert, M., Zambrano, D., Schönhuth, A. and Bohte, S., 2018. An image representation based convolutional network for DNA classification. arXiv preprint arXiv:1806.04931.

[3] Lopez-Rincon, A., Tonda, A., Mendoza-Maldonado, L., Claassen, E., Garssen, J. and Kraneveld, A.D., 2020. Accurate identification of sars-cov-2 from viral genome sequences using deep learning. bioRxiv.

[4] Phan, D., Ngoc, G.N., Lumbanraja, F.R., Faisal, M.R., Abipihi, B., Purnama, B., Delimiyanti, M.K., Kubo, M. and Satou, K., 2017. Combined Use of k-Mer Numerical Features and Position-Specific Categorical Features in Fixed-Length DNA Sequence Classification. Journal of Biomedical Science and Engineering, 10(8), pp.390-401.

[5] Lopez-Rincon, A., Tonda, A., Mendoza-Maldonado, L., Mulders, D.G., Molenkamp, R., Perez-Romero, C.A., Claassen, E., Garssen, J. and Kraneveld, A.D., 2021. Classification and specific primer design for accurate detection of SARS-CoV-2 using deep learning. Scientific reports, 11(1), pp.1-11.

[6] Kassim, N.A. and Abdullah, A., 2017. Classification of DNA sequences using convolutional neural network approach. Innovations in Computing Technology and Applications, 2, pp.1-6.

[7] Leung, KwongSak, et al. "Data mining on dna sequences of hepatitis b virus." IEEE/ACM transactions on computational biology and bioinformatics 8.2 (2011): 428-440.

[8] Whata, A. and Chimedza, C., 2021. Deep Learning for SARS COV-2 Genome Sequences. IEEE Access, 9, pp.59597-59611.

[9] Zhang, X., Beinke, B., Kindhi, B.A. and Wiering, M., 2020. Comparing Machine Learning Algorithms with or without Feature Extraction for DNA Classification. arXiv preprint arXiv:2011.00485.

[10] Bosco, G.L. and Di Gangi, M.A., 2016, December. Deep learning architectures for DNA sequence classification. In International Workshop on Fuzzy Logic and Applications (pp. 162-171). Springer, Cham.

[11] Rizzo, R., Fiannaca, A., La Rosa, M. and Urso, A., 2015, September. A deep learning approach to dna sequence classification. In International Meeting on Computational Intelligence Methods for Bioinformatics and Biostatistics (pp. 129-140). Springer, Cham.

[12] Gunasekaran, H., Ramalakshmi, K., Rex Macedo Arokiaraj, A., Deepa Kanmani, S., Venkatesan, C. and Suresh Gnana Dhas, C., 2021. Analysis of DNA Sequence Classification Using CNN and Hybrid Models. Computational and Mathematical Methods in Medicine, 2021.

[13] Akkaya, U.M. and Kalkan, H., 2021, October. Classification of DNA Sequences with k-mers Based Vector Representations. In 2021 Innovations in Intelligent Systems and Applications Conference (ASYU) (pp. 1-5). IEEE.

[14] Mahmoud, M.A. and Guo, P., 2021. DNA sequence classification based on MLP with PILAE algorithm. Soft Computing, 25(5), pp.4003-4014.

[15] Higashihara, M.A.S.A.N.O.R.I., Rebolledo-Mendez, J.D., Yamada, Y.O.I.C.H.I. and Satou, K.E.N.J.I., 2008. Application of a feature selection method to nucleosome data: accuracy improvement and comparison with other methods. WSEAS Transactions on Biology and Biomedicine, 5(5), pp.95-104.

[16] Mikolov, T., Chen, K., Corrado, G., & Dean, J. (2013). Efficient estimation of word representations in vector space. arXiv preprint arXiv:1301.3781.

[17] Mikolov, T., Sutskever, I., Chen, K., Corrado, G. S., & Dean, J. (2013). Distributed representations of words and phrases and their compositionality. In Advances in neural information processing systems (pp. 3111-3119).

[18] Devlin, J., Chang, M. W., Lee, K., & Toutanova, K. (2018). Bert: Pre-training of deep bidirectional transformers for language understanding. arXiv preprint arXiv:1810.04805.

[19] Albawi, S., Mohammed, T. A., & Al-Zawi, S. (2017, August). Understanding of a convolutional neural network. In 2017 International Conference on Engineering and Technology (ICET) (pp. 1-6). Ieee.

[20] Graves, A., Fernández, S., & Schmidhuber, J. (2005, September). Bidirectional LSTM networks for improved phoneme classification and recognition. In International conference on artificial neural networks (pp. 799-804). Springer, Berlin, Heidelberg.

[21] Lee, S., & Lee, C. (2020). Revisiting spatial dropout for regularizing convolutional neural networks. Multimedia Tools and Applications, 79(45), 34195-34207.

[22] Baldi, P., & Sadowski, P. J. (2013). Understanding dropout. Advances in neural information processing systems, 26, 2814-2822.

[23] Agarap, A. F. (2018). Deep learning using rectified linear units (relu). arXiv preprint arXiv:1803.08375.

[24] Prechelt, L. (1998). Early stopping-but when?. In Neural Networks: Tricks of the trade (pp. 55-69). Springer, Berlin, Heidelberg.

[25] Cho, K., Van Merriënboer, B., Bahdanau, D., & Bengio, Y. (2014). On the properties of neural machine translation: Encoder-decoder approaches. arXiv preprint arXiv:1409.1259.