mathematics

MDPI

Article

Bio-Constrained Codes with Neural Network for Density-Based DNA Data Storage

Abdur Rasool [1,2], Qiang Qu [1], Yang Wang [1] and Qingshan Jiang [1,*]

[1] Shenzhen Key Laboratory for High Performance Data Mining, Shenzhen Institute of Advanced Technology, Chinese Academy of Sciences, Shenzhen 518055, China; rasool@siat.ac.cn (A.R.); qiang@siat.ac.cn (Q.Q.); yang.wang1@siat.ac.cn (Y.W.)

[2] Shenzhen College of Advanced Technology, University of Chinese Academy of Sciences, Shenzhen 518055, China

* Correspondence: qs.jiang@siat.ac.cn; Tel.: +86-755-8639-2340

Abstract: DNA has evolved as a cutting-edge medium for digital information storage due to its extremely high density and durable preservation to accommodate the data explosion. However, the strings of DNA are prone to errors during the hybridization process. In addition, DNA synthesis and sequences come with a cost that depends on the number of nucleotides present. An efficient model to store a large amount of data in a small number of nucleotides is essential, and it must control the hybridization errors among the base pairs. In this paper, a novel computational model is presented to design large DNA libraries of oligonucleotides. It is established by integrating a neural network (NN) with combinatorial biological constraints, including constant GC-content and satisfying Hamming distance and reverse-complement constraints. We develop a simple and efficient implementation of NNs to produce the optimal DNA codes, which opens the door to applying neural networks for DNA-based data storage. Further, the combinatorial bio-constraints are introduced to improve the lower bounds and to avoid the occurrence of errors in the DNA codes. Our goal is to compute large DNA codes in shorter sequences, which should avoid non-specific hybridization errors by satisfying the bio-constrained coding. The proposed model yields a significant improvement in the DNA library by explicitly constructing larger codes than the prior published codes.

Keywords: DNA data storage; bio-constrained codes; neural network; DNA computing

MSC: 68U35

Citation: Rasool, A.; Qu, Q.; Wang, Y.; Jiang, Q. Bio-Constrained Codes with Neural Network for Density-Based DNA Data Storage. *Mathematics* **2022**, *10*, 845. https://doi.org/10.3390/math10050845

Academic Editor: Danny Barash

Received: 31 January 2022
Accepted: 1 March 2022
Published: 7 March 2022

1. Introduction

The exponential increase in big data demands high density and capacity storage. Inspired by nature, DNA (deoxyribonucleic acid) has various applicable features for digital data storage. DNA comprises four bases: adenine (A), guanine (G), cytosine (C), and thymine (T), collectively called nucleotides. DNA data storage has three key steps [1–7]: (i) Digital data are converted into binary data, which are encoded into DNA strands with quaternary alphabet (A, C, T, and G) strings/sequences that are called DNA codes or codewords. (ii) These strands are synthesized (data writing) into oligonucleotides by a DNA synthesizer, and the data are stored. (iii) DNA strands are decoded by DNA sequencing (data reading) to retrieve the data. These key steps come under the big umbrella of DNA computing, in which DNA data storage is partially based on information technology (IT) and biotechnology (BT). In IT, data encoding and decoding techniques are employed, comprising computational and mathematical models. In BT, DNA synthesis, storage, and sequencing are carried out with base pairs (A, C, G, and T) in a DNA molecule. It is essential for any DNA computing model to select the DNA molecules and code them efficiently to attain maximum storage density [8].

In the DNA data storage system, various coding techniques, i.e., biological constraint/bio-constraint coding [9] and error correction coding [5], are presented to overcome the different ambiguities. In the literature, GC-content [3], no-run-length [10], reverse-complement (RC) constraint [11], and Hamming distance (d_H) have been found as major biological constraints coding for DNA synthesis and sequencing errors. For any DNA sequence $\alpha = \alpha_1\alpha_2 \ldots \alpha_n \in \sum^n$, the GC-content is the ratio of the sum of bases content (G and C) to the total number of bases ($|G + C|/|n|$) $\times$ 100% for n sequence length. It is close to 50% in each DNA codeword [12]. No-run-length is the avoidance of repetition of the same quaternary (q-ary) alphabet. Similarly, for the RC constraint, the reverse sequence $\alpha^r = \alpha_n\alpha_{n-1} \ldots \alpha_1$, complement sequence $\alpha^r = \alpha_1^c\alpha_2^c \ldots \alpha_n^c$, and reverse-complement sequence are $\alpha^{rc} = \alpha_n^c\alpha_{n-1}^c \ldots \alpha_1^c$, for which $A^C = T$, $T^C = A$, $G^C = C$, and $C^C = G$. For instance, in a given DNA sequence TTCAGGA, the reverse is ATGACGT, the complement is AAGTCCT, and the reverse-complement is TACTGCA. $A_4^{GC}(n, d, \omega)$ and $A_4^{GC,RC}(n, d, \omega)$ denote the maximum number of codewords in a DNA code satisfying two constraints (GC-content and d_H) and satisfying three constraints (GC-content, RC constraint and d_H), respectively. DNA libraries satisfying these bio-constraints will have a certain application to DNA computing, particularly DNA data storage [3,10].

A DNA sequence is read through a particular hybridization process in which two complementary single-stranded DNA molecules are combined to form a single-stranded molecule via base pairing. If any sequence in DNA codewords is identical to its reverse-complement, non-specific hybridization will occur, which causes the leading errors in retrieving information [2]. To avoid these, the authors utilized stochastic-based optimization algorithms and neural networks [13–17]. The algorithms in [13,17] were introduced to improve the lower bounds of DNA codes by different iterations and parameters.

Recently, a deep learning model (DLM) was introduced with three different next-generation sequences for DNA information storage [14]. Their DLM utilized four gated recurrent unit (GRU) neural networks grouped into two sets, which took sequences from 5′ to 3′ or from 3′ to 5′. GRUs with two gates have been adopted with a feed-forward neural network (FFNN) to predict the sequence during the hybridization process. The excessive number of hidden nodes and the model's reliance on an FFNN exacerbate the DLM training. Despite this issue, it initiated an acceptive idea to bring neural networks for DNA data storage. This milestone has motivated us to implement a neural network on DNA codes to provide a novel coding scheme for high-density storage. In [15], a DeepMod system was proposed that integrates the recurrent neural network and long short-term memory (LSTM) models to perceive the DNA codes from various Oxford Nanopore sequences. Likewise, a convolutional neural network was embedded for the generation of DNA bases to achieve high-density storage. It employed a DNA-mapping method consisting of GC-content and homopolymer length constraints to design the DNA codebook [16]. Their work inspired us to seek a model that generalizes the DNA codewords as much as possible with the artificial neural network. However, the neural network (NN) with a fixed number of nodes, i.e., LSTM including forwarding pass, is not well-studied for DNA sequence input. Apart from non-specific hybridization, it is essential for DNA code development to detect the error source to avoid insertion, deletion, and substitution errors with DNA coding bio-constraints [4,5,12]. Several existing studies have been conducted to address these problems with DNA codewords. However, the literature that satisfies the GC-content, Hamming distance, and RC constraints is cited here.

In 2004, DNA codes with GC-content were extensively presented in [18]. It reported the upper and lower bounds on DNA code size with GC-content and Hamming distance to construct the DNA codewords. In a polymerase chain reaction (PCR), a DNA code with huge GC-content (say, >60%) caused the insertion and deletion errors. Thus, it is necessary to consider the GC-content for the stability of DNA sequences by avoiding computation errors. In 2017, a study pioneered by Erlich [3] delivered a seminal work on DNA data storage by proposing a fountain code with GC-content (45–55%) and a minimum Hamming distance (d). They achieved 1.57 net information density; however, they still faced errors

in GC-content which propagated severe errors, including mismatches, deletions, and insertions, during the decoding process. In addition, no theoretical lower or upper bounds were presented for those constraints.

In 2018, ref. [10] proposed a novel altruistic algorithm with lower bounds to generate constraint-based stable DNA codes. It also used constant GC-content and minimum Hamming distance and reported an improved number of DNA codewords. However, the storage efficiency was not sufficiently considered for density-based DNA data storage. In 2020, the author [17] proposed a damping multi-verse optimizer algorithm to design the DNA coding sets by constructing the GC-content with no-run-length constraints. Their results revealed 4–16% more improved DNA coding than that of [10], which suggests that the increase in constraints can improve the codes for high-density DNA data storage. In 2021, our previous paper [12] extended the work [10] by proposing a novel algorithm to construct the DNA coding sets with improved lower bounds. The proposed algorithm was applied with GC-content and no-run-length constraints and achieved 30% better lower bounds. However, besides the insertion and deletion errors in DNA codes, another issue of secondary structures (SS) occurs during the reading process [19]. The SS is a base pairing contact of a single-strand sequence that folds back on itself, as presented in [11] (Figure 1). Any DNA sequence with an SS shape will consume the extra resources and energy to be unfolded, which slows the chemical reaction immensely. Therefore, DNA needs to be free from the SS shape before reading DNA sequences in the wet lab. There are few studies on eliminating this severe issue. The author in [11,20] introduced the RC constraint to overcome the SS issue. They subjected the GC-content and RC-content together to improve the DNA coding sets. Their studies furnish the basic idea of combinatorial constraints to generate DNA codes with minimum errors. Although the literature mentioned above [10–12,17,20] received high storage DNA code sets and coding rates, these studies do not provide a sufficient method to design higher DNA codes in the shorter sequences that must satisfy the biological constraints, which is enormously important for a stable density-based DNA storage system.

This paper introduces a more efficient coding technique with a novel computational model that is based on biologically inspired computing because it uses a neural network (NN) with biological constraints to obtain a high-density-based DNA data storage. In the proposed model, LSTM as an NN with a forward pass is utilized to open a new door in the NN for DNA code construction. Firstly, the binary data are converted into premiere DNA bases by using the [3] scheme. Then, the yielded premiere DNA strings are passed through the NN model with the forward passing mechanism. A particular criterion trains the activation functions to randomly generate DNA codes. If those DNA codes pass that criterion, we term them optimal DNA codes. Then, the combinatorial constraints are utilized to concatenate these optimal DNA codes. The combinatorial constraints, including GC-content and RC constraint with Hamming distance, are computed to generate a DNA library that is used to store the digital information, for which different propositions and theorems are constructed in the Magma program and proved in this paper. GC-content and Hamming distance are computed, and results are obtained with improved lower bounds. Meanwhile, the RC constraint with Hamming distance is constructed to avoid secondary structure, and it is concatenated with GC to generate the DNA library with the best-known codes. These codes are generated by Magma with different inequalities. These inequalities are based on the previous studies that are used for the comparison of our results. Furthermore, the results are analyzed by the coding rate formula, which helps us to evaluate the data storage density in DNA media.

In general, there are two goals to be delivered for high-density-based DNA data storage with the following features:

1. To improve the net information density by storing a large amount of digital data in shorter DNA sequences.
2. To construct the DNA codes that satisfy the combinatorial bio-constraints to overcome the reading errors.

In this scenario, these goals are accomplished by the following significant contributions:

- A novel computational model based on the LSTM neural network with a forward pass is proposed to generate the optimal DNA codes from the premiere DNA bases. To the best of our knowledge, such a model has not been studied in the prior studies.
- The combinatorial bio-constraints, including GC-content, RC constraint, and Hamming distance, are constructed for optimal DNA codes to avoid non-specific hybridization by overcoming sequencing errors and secondary structures.
- The results receive many DNA coding sets satisfying the bio-constraints and significantly improving the DNA coding rates compared to the existing studies.

The structure of the rest of the paper is as follows: Section 2 delivers the prior work about deep neural network and combinatorial constraints for DNA data storage. Section 3 presents the preliminaries and notations, Section 4 introduces the proposed model, Section 5 elaborates on the results, and Section 6 concludes this work.

2. Literature Review

This section is divided into two subsections to emphasize our paper's contributions based on neural networks for DNA codes and DNA coding with combinatorial constraints.

2.1. Deep Neural Networks for DNA Codes

DNA computing has successfully impacted human life due to well-known computation tools based on machine learning and the deep learning community. With the rapid generation of digital data, efficient and effective deep learning architectures (DLAs) have been constructed to compute big data [21]. DLA has been approved in a variety of domains with significant accuracies and predictions. In this article, we consider applying the deep neural network based on DLA. Recurrent neural networks (RNNs) provide the connection between the nodes to form a directed graph along a temporal sequence. The graph exhibits a short-term memory that allows RNNs to remember information from the previous state to the next state [22]. Long short-term memory (LSTM) is a variant of RNN that efficiently learns the long-term dependencies. It has three gates: input, output, and the forget gate. An LSTM unit has a node or cell that accounts for the values over particular time intervals while the rest of the gates regulate the information [23,24].

Various deep neural networks have been applied with different methods and models in various natural computing studies. In 2015, a novel method was proposed for the transformation of DNA sequences into numerical sequences. This method was based on a pulse-coupled neural network and Huffman coding, which used triplet codes to encode the different lengths of DNA sequences [25]. Another study also attempted to encode the data with that method, but it found that encoded sequences are compressed at a close distance, making the results less informative [26]. Although numerous studies have been conducted with deep neural networks for natural computing, deep neural networks are still new models for the DNA data storage system. For instance, in 2021, a GRU (gated recurrent unit)-based deep learning model was presented for DNA information storage with next-generation sequence prediction [14]. Similarly, the author proposed a DeepMod system that integrates the RNN and LSTM models to perceive the DNA codes from various Oxford Nanopore sequences. While RNNs are employed to capture the Nanopore sequencing, LSTM overcomes the vanishing gradient issues in the training of RNNs. The proposed system collectively achieves better DNA codes from the given sequences compared to others [15]. In addition, in [16], the problem of DNA synthesis cost was addressed by delivering a high-density-based DNA data storage system. To achieve a high storage density, convolutional neural networks were embedded to generate the DNA bases. It employed a DNA-mapping method that consisted of GC-content and homopolymer length constraints to design the DNA codebook. It was reported that the proposed scheme efficiently stored and retrieved the information from the DNA storage system with the integration of a deep neural network with the DNA mapping method.

These studies provide the motivation for the integrational system of deep neural networks with DNA coding and combinatorial bio-constraints.

2.2. DNA Coding with Combinatorial Bio-Constraints

In DNA synthesizing and sequencing, various errors occur, which are combated with different coding techniques. For instance, error correction coding and bio-constraint coding are mainly used in DNA-based information storage systems [27]. Bio-constraint coding has been practically applied in mass data storage, i.e., magnetic and optical data recording [28]. There are different types of DNA constraint coding reported in the existing literature. Researchers [2,29,30] have formulized single constraints and/or combined the constraints to attain the targeted results by preventing DNA sequence (DNA code) errors.

G. M. Church delivered a pivotal work on DNA data storage by converting 8-bit ASCII (0 to A or C and 1 to G or T) into DNA bases. It pondered the GC-content constraint and the homopolymer run-length constraint by disallowing a run-length greater than 3. It was a groundbreaking step toward storing the digital information into DNA, but it was plagued by huge errors and a lack of competency [7]. N. Goldman presented a different method by compressing the raw data into DNA sequencing with differential coding and a Huffman coding scheme. It employed the run-length constraint of at most 1 and achieved an effective coding rate. It suggested considering the GC-content for better constraint satisfaction [6]. Similarly, R. N. Grass combined the different constraints to deliver an error-correcting scheme. It used Reed–Solomon codes for error control [5]. In comparison, M. Blawat offered a seminal study for DNA data storage by proposing a forward error-correction mechanism. It provided the codewords (codes) that avoided deletion and substitution errors by utilizing the GC-content constraint [4]. Y. Erlich and D. Zielinski proposed another benchmark study by designing the fountain code that considered the GC-content and run-length for DNA synthesis and sequencing. Their study significantly achieved better coding rates compared to existing work; however, in the decoding process, it found error propagation in the information retrieval stage [3]. W. Song and Y. Wang also conducted research on DNA data storage by presenting a mathematical method for DNA code generation by preserving GC-content and Hamming distance constraints [9,27]. D. Limbachiya constructed an altruistic algorithm to create DNA codewords of a specific length. The algorithm also formalized the Hamming distance for each code to satisfy the constraints [10]. In our previous work, a novel algorithm was developed to generate the DNA codes. The obtained DNA codes' errors have been corrected to a limited extent with GC-content and no-run-length constraints [12].

In conjunction with GC-content, no-run-length, and Hamming distance, a few inspiring and influential constraint studies deal with the reverse-complement constraint. In 2005, Oliver D. King reported the theoretical lower and upper bounds for the maximum size of DNA codes. The report created the codes with the minimum Hamming distance and reverse-complement of any code with the least distance. It stated that obtained DNA codes were larger than ever [18]. In 2010, A. Niema accommodated Oliver D. King's bounds to design DNA codewords with GC-content, the Hamming distance, and the reverse-complement RC constraint by avoiding non-specific hybridizations. They employed RC constraint to handle the searching of codes related to bases with 0 and 1 points. It obtained many new codes for DNA data storage [20]. In 2021, Benerjee K.G. exhibited the families of those DNA codes which avoid secondary structures. It combined the RC constraint with homopolymers' run-lengths to construct the dissimilar DNA codes [11].

The prior work on bio-constrained coding established an idea of combinatorial constraints with different mathematical methods and formulations. These studies achieved many DNA codes in their particular methods. However, we have found that the desired DNA codes are still capable of improving the lower and upper bounds for the generation of high-density-based DNA codes. As we discuss a few studies on deep neural networks that impact constructing DNA codes, in this paper, we propose a novel model that integrates a deep neural network with combinatorial constraints to design DNA codewords.

3. Preliminaries and Notations

According to fundamental bio-constraints (Section 1), we design the oligos as sequence α for $\sum = \{A, C, G, T\}$. If $\alpha \in \sum^n$, the alphabet at the position i in the sequence α is presented as α_i. Thus, a sequence $\alpha_i = \alpha_1\alpha_2 \ldots \alpha_n \in \sum^n$ will be generated. In the same way, let another sequence $\beta_i = \beta_1\beta_2 \ldots \beta_n, \in \sum^n$ be possible if the Hamming distance [31] between both sequences $(\alpha, \beta \in \sum^n)$, denoted $d_H(\alpha, \beta)$, satisfies the following:

$$d_H(\alpha, \beta) = |\{1 \leq i \leq n : \alpha \neq \beta\}|. \quad (1)$$

Apart from $d_H(\alpha, \beta)$, sequences $\alpha, \beta, \in \sum^n$ must satisfy the GC-content and reverse-complement constraints (Section 3) to produce the DNA library $\mathcal{L}$. Hereafter, oligos are denoted by Greek letters, and other notations are a generic set ξ of sequences $\alpha, \beta, \in \sum^n$. Here, we need to provide the definition of the DNA library [31].

Definition 1. *A set of DNA bases {A,C,G,T} with n-mer oligos $\xi \subseteq \sum^n$ that satisfies the constant GC constraint, reverse-complement constraint, and Hamming distance constraint is called a DNA code/codeword (n, d, ω) which collectively forms a DNA library $\mathcal{L} = \sum(n, d, \omega)$.*

If a DNA library is indicated by the number of K-constrained sequences of $q - ary$ strings initiated with a non-zero symbol i, Shannon's relationship [32] can be written as a recurrent relationship:

$$\mathcal{L}_k(n) = (q-1)\sum_{i-1}^{K+1} \mathcal{L}_k(n-1), \; n > K. \quad (2)$$

If the number of codes n increases, the $\mathcal{L}_k(n)$ increase exponentially by the following:

$$\mathcal{L}_k(n) \sim c\Gamma_k^n, \quad n >> 1, \quad (3)$$

where $c \sim 1$ is a constant and Γ_k is an exponential growth factor which is a real root of

$$\Gamma_K^{K+2} - q\Gamma_K^{K+1} + q - 1 = 0. \quad (4)$$

In order to store the digital data in the nucleotide, this expression leads to defining the DNA data density [32].

Definition 2. *The maximum number of digital data bits (b) stored per nucleotide (nt) is termed as data density, denoted by D_k and defined:*

$$D_k = \lim_{n\to\infty}\left(\frac{1}{n}\log_2 \mathcal{L}_k(n)\right) = \log_2 \Gamma_k \left(\frac{b}{nt}\right). \quad (5)$$

In high-density-based DNA storage, there is a probability of secondary structures. In experiments, the Nussinov–Jacobson (NJ) algorithm is employed to predict the secondary structures approximately [33]. During the chemical reaction, a DNA sequence $\alpha_i = \alpha_1, \alpha_2, \ldots, \alpha_n$ releases the energy to attain stability after forming secondary structures. Thus, this form can be calculated by a DNA property called free energy (E). This energy relies on the sequence pair $\left(\alpha_i, \beta_j\right)$, where $1 \leq i < j \leq n$ and the pair releases its energy, which is termed as interaction energy $\varphi\left(\alpha_i, \beta_j\right)$. Note that $\varphi\left(\alpha_i, \beta_j\right)$ between sequence α_i and β_j in any pair $\left(\alpha_i, \beta_j\right)$ will be independent of other sequence pairs. In the NJ algorithm, the interaction energies depend on the selected sequence pairs $\left(\alpha_i, \beta_j\right)$ as non-positive values, while, for the independent interaction energies, the assumption for the NJ algorithm with minimum free energy $E_{i,j}$ for a DNA sequence $\alpha_i = \alpha_1\alpha_2 \ldots \alpha_n$ is

$E_{i,j} = \min\left\{E_{i+1,j-1} + \varphi\left(\alpha_i,\ \beta_j\right),\ E_{i,k-1} + E_{k,j} : K = i+1, \ldots, j\right\}$ for particular conditions and $E_{l,l} = E_{l-1,l} = 0$ for $l = 1, 2, \ldots, n$ [34].

4. Proposed Model

In this paper, the concept of a neural network is embedded with the combinatory constraints $A_4^{GC,RC}(n,\ d,\ \omega)$ to design the DNA codes $(n,\ d,\ \omega)$ of nucleotides that preserve the GC-content, Hamming distance $d_H(\alpha,\ \beta)$, and RC constraints. The proposed model is built on the three layers listed below:

1. Transform the digital data into the sequence of bases (A, C, G, and T).
2. Encode the DNA bases into optimal DNA codes.
3. Create the bio-constraint codes for the DNA library construction.

This paper's model novelty is based on layers 2 and 3. In contrast, the first layer is described in prior literature [3], in which, firstly, a digital data file is compressed into binary format. In our case, we compressed an image (cat.jpg) file into a binary file. Next, the binary file is preprocessed with different segments. Furthermore, it reiterates 2 computation processes: the Luby transform and screening. In the Luby transform, different bases are sets for fountain codes, while screening translates the binary droplet to a DNA sequence by converting (00, 01, 10, and 11) to (A, C, G, and T), respectively. Thereafter, we will term these codes premiere codes $\xi \subseteq \sum^n$.

After encoding, bio-constraints are applied to screen the sequences. However, in layer 2, we intend to apply the following neural network (NN) over the encoded premiere codes $\xi \subseteq \sum^n$ to generate optimal DNA codes Φ_{DNA} for high-density storage. The 3rd layer introduces the bio-constrained coding to overcome the errors and construct the DNA library $\mathcal{L} = \sum(n,\ d,\ \omega)$. The comprehensive details of these layers (2 and 3) are given in the following subsections. The model diagram for the integration of NN with DNA coding constraints is illustrated in Figure 1.

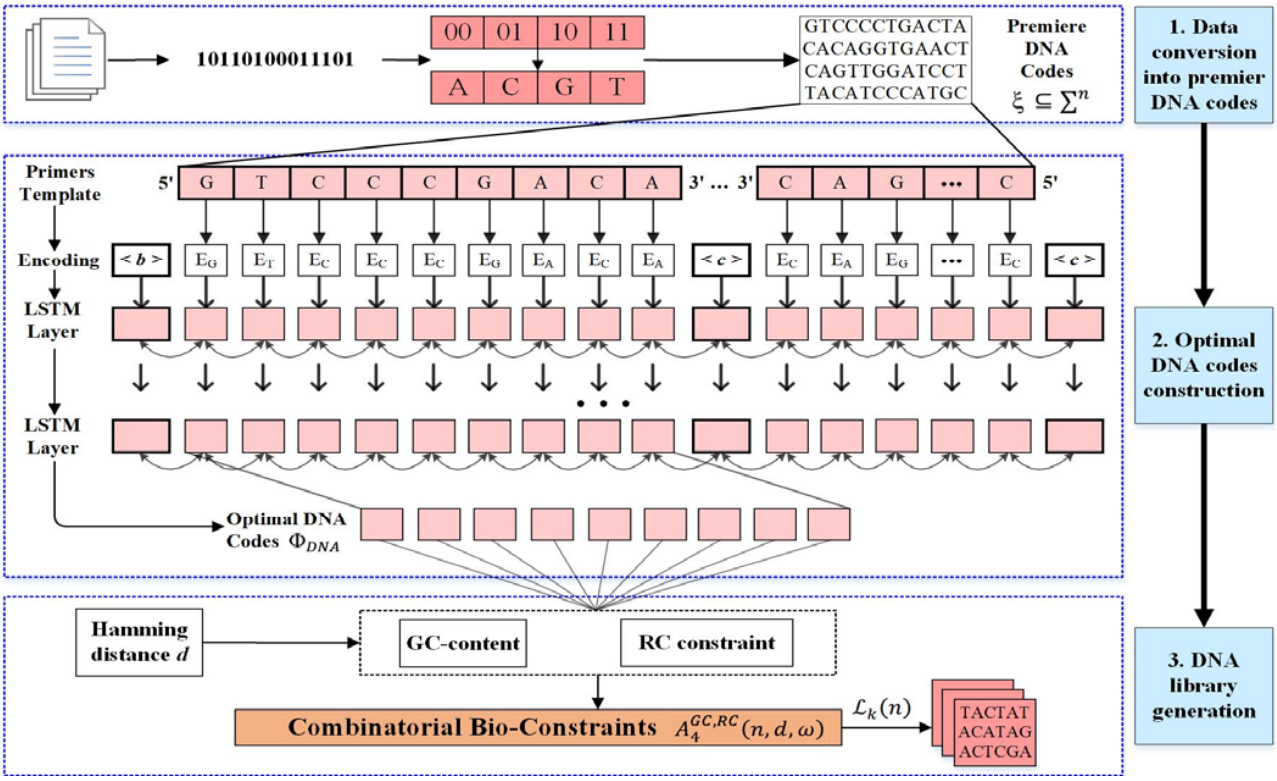

Figure 1. The proposed computational model with NN and combinatorial bio-constraints for DNA data storage.

4.1. NN-Based DNA Codes

The encoded premiere DNA codes $\xi \subseteq \sum^n$ are moved through the NN model to obtain the optimal DNA codes Φ_{DNA}. This model is based on 4 layers (encodings, 2 LSTM layers, and forward pass). In this neural network, 128 LSTM units have been considered, while the amount of hidden units is 4 times greater than that of the LSTM units and the dropout rate is set to 0.5 to avoid the overfitting issue. This rate will result in a 50% decrease in the number of neurons in the repetition oligonucleotides. It creates weight = 0 if a code is in a forward pass for a single iteration. During the training process, the trainable parameter is automatically set to false to prevent weight updates. The model is trained on forward and reverse input sequences $(\alpha,\ \beta)$ to append the DNA codes, which must have different oligos in front of each other. In this paper, various primer templates are created from the $\xi \subseteq \sum^n$ with length 9 bases to make model learning efficient. The learning sequences are essential to attain the DNA codewords, avoiding identical bases. A sequence $\alpha_i = \alpha_1\alpha_2 \ldots \alpha_n$, learned from one layer, is concatenated to another layer according to the forward pass mechanism. In the encoding layer, two single-stranded sequences $(\alpha,\ \beta)$ are concatenated by inserting a particular connector $< c >$ token, which serves as an ending token also. In addition, another special token $< b >$ is appended at the beginning of the sequence. Each encoded base E_i is indexed and fed through the LSTM layers. These layers are double stacked, and the unique tokens are transferred through the dense layers. Each sequence α or β in these layers interacts with two-headed arrows to present the bi-directional LSTM for readability. All sequence nodes are initiated from 0 and updated based on the next nucleotide's information. The hidden LSTM nodes predict the potential patterns and the forget gates update all nucleotides in the given DNA sequence. In the last layer, the final sequence updating is passed through the forward pass of LSTM to identify the forward-base DNA code Φ_{DNA}.

To construct the NN model that permits the constant flow of sequences $(\alpha,\ \beta)$ through self-connected units, each oligonucleotide is protected with a self-linear unit j. The input gate in_j unit is responsible for protecting the linear unit j from the other irrelevant units' connection. Next, the critical unit, a memory cell, is designed for each DNA sequence with a linear unit j to stop connecting with different DNA sequences. The memory cell of j unit is indicated by c_j, with the current net sequence net_{c_j} with c_j achieving the input from the multiplicative out_j unit, which is considered the output gate in the LSTM model. The activation of the input gate in_j and output gate out_j with iteration time t is indicated by $y^{in_j}(t)$ and $y^{out_j}(t)$, respectively, which can be defined [23]:

$$y^{in_j}(t) \ = \ f_{in_j}\Big(net_{in_j}(t)\Big), \tag{6}$$

where

$$net_{in_j}(t) \ = \ \sum_u w_{in_j u} y^u(t-1).$$

$$y^{out_j}(t) \ = \ f_{out_j}\Big(net_{out_j}(t)\Big), \tag{7}$$

where

$$net_{out_j}(t) \ = \ \sum_u w_{out_j u} y^u(t-1).$$

where w stands for the number of the weight matrix and y^u is the activation of an arbitrary unit u.

These activation functions enable the network to learn the complex features of each DNA sequence at the input gate in_j and output gate out_j. Although there are other weights and vector formulas for the LSTM gates [35,36], we omit them in this paper. However, we generalize the above activation functions to architect the forward pass for the final output. These functions learn the DNA bases to satisfy the following criteria for primer design.

- The DNA primer length is generally 15–30 nt [6]. The best length for PCR amplification primers is usually 20 nt; we also train our model at this limit.

- The length of repeated bases in a primer is generally $\leq$4 nt [5]. The consecutive appearance of any particular base makes the unstable DNA structure. We set consecutive base lengths ≤ 3 nt.
- The GC ratio of the primers should be 45–55% [3]. The bases A and T are linked by 2 hydrogen bonds, and C and G bases are connected by 3 hydrogen bonds (see Figure 6 [37]). We also consider the GC-content to be 45–55% in this work.

If the primer does not satisfy the above criteria, we alter one base from one primer. For example, if a primer does not satisfy the condition with the sequence AGGTCATC, we alter the first base 'A' with 'T', because they have a connected hydrogen bond between them, and reconfirm the criterion, while, if the primer meets the criteria, the premiere DNA codes $\xi \subseteq \sum^n$ are trained to the multiplicative units. The j-th memory cell block c_j, which found the input from multiplicative units in_j and out_j, will have a v-th unit of a memory cell block c_j^v for a net input $net_{c_j^v}$ for time t:

$$net_{c_j^v}(t) \;=\; \sum_u w_{c_j^v u} y^u(t-1). \tag{8}$$

The internal state $s_{c_j^v}$ and output activation $y^{c_j^v}$ of the v-th memory for a time t with memory cell block c_j will be:

$$s_{c_j^v}(t) \;=\; s_{c_j^v}(t-1) + y^{in_j}(t) g\Big(net_{c_j^v}(t)\Big). \tag{9}$$

$$y^{c_j^v}(t) \;=\; y^{out_j}(t) h\Big(s_{c_j^v}(t)\Big). \tag{10}$$

The final net input for the index k, which ranges for output units and ranges of the final activation of output with t are:

$$y^k(t) \;=\; f_k(net_k(t)), \tag{11}$$

where

$$net_k(t) \;=\; \sum_{u:\ u \text{ not a gate}} w_{ku} y^u(t-1).$$

Note that each memory cell has its weight w for the final net input $net_k(t)$. The DNA sequence is updated with the latest bases to design the DNA library. Finally, the LSTM cell determines the output by assigning these updates to the output gate out_j. The out_j computes the final output activation $y^{c_j^v}$ that is passed through the cell as a final optimal DNA code Φ_{DNA}.

4.2. Combinatorial Constraints

This section deliberates the coding method to map the optimal DNA codes Φ_{DNA} with sequence length $k(2n-1)$ to the DNA library $\mathcal{L}_k(n)$ with sequence length kn that satisfies the combinatorial constraints (GC, $d_H(\alpha, \beta)$, RC). The basic idea is to combine or concatenate the k optimal sequences (α, β) of length n to the sequence of length kn by constructing adjacency relations. For instance, if $\alpha_i = \alpha_1\alpha_2 \ldots \alpha_n$ and $\beta_i = \beta_1\beta_2 \ldots \beta_n$ are sequences with length n, then $\alpha_i\beta_i = \alpha_1\alpha_2 \ldots \alpha_{kn}\beta_1\beta_2 \ldots \beta_{kn}$ is the concatenation of α_i and β_i. Since the prescribed parameters of k and $n > 3$ to guarantee the sequence α is optimal are met, it is necessary that $\alpha'_{i-1}\alpha_i$ will be an optimal sequence, if $\forall\, i \in \{2, 3, \ldots, k\}$, where α'_{i-1} indicates the sequence which must have 3 symbols of α_{i-1}.

The constant GC-content ω can be presented analogously as $A_4^{GC}(n, d, \omega)$ for the $A_4(n, d, \omega)$ if all DNA codewords in Φ_{DNA} have similar melting temperatures and each code desires to be ω. The following are the upper and lower bound constraints for the DNA library $\mathcal{L}_k(n)$ construction. Proposition 1 is based on upper bounds with modified variables for the Hamming distance d, while the original proposition [18] considered only

the number of sequences n. Due to new variables, the proof is presented with new codes for this work.

Proposition 1. *For the sequences* $(\alpha,\ \beta)$ *having the number of codewords* $n > 0$*, with constraint* $0 \leq d \leq n$ *and* $0 \leq \omega \leq n$ *for the upper bound,*

$$A_4^{GC}(n,\ d,\ \omega) = \begin{cases} 2\ if\ \omega < \frac{n}{3}\ or\ \omega > \frac{2d}{3} \\ 3\ if \frac{n}{3} \leq \omega < \frac{d}{2}\ or\ \frac{n}{2} < \omega \leq \frac{2d}{3} \\ 4\ if\ \omega = \frac{n}{2} \end{cases} \tag{12}$$

Proof. Say that if there are 3 codewords having GC-content $\omega < \frac{n}{3}$, there will be some position i where none of the words has C or G; thus, 2 of 3 words should agree in that position. Hence, $A_4^{GC}(n,\ d,\ \omega) \leq 2$ and if $\omega < \frac{n}{3}$, then 2 codewords will be $C^{\omega}A^{n-\omega}$ and $G^{\omega}T^{n-\omega}$. In contrast, if there are 4 codewords and none of the codes is agreed at any position i, then all 4 nucleotides will occur in each position of i. Thus, the average GC-content will be $\frac{n}{2}$, which is based on $A_4^{GC}(n,\ d,\ \omega) \leq 3$, and 3 codewords with $\frac{n}{3} \leq \omega < \frac{n}{2}$ will be $C^{\omega}A^{n-\omega}$, $T^{n-\omega}C^{\omega}$, and $A^{\lfloor \frac{n-\omega}{2} \rfloor}G^{\omega}T^{\lfloor \frac{n-\omega}{2} \rfloor}$. Similarly, if there are 2 codewords and no agreement in any position, there can be 4 codes according to the pigeonhole principle. Thus, the average GC-content will be ω for $A_4^{GC}(n,\ d,\ \omega)\ \leq 4$ and 4 codewords will be $A^{\omega}C^{\omega}$, $C^{\omega}A^{\omega}$, $T^{\omega}G^{\omega}$, and $G^{\omega}T^{\omega}$. □

From this proposition, Theorem 1 is derived by considering the $n-1$ code length to generate the improved DNA coding sets. In contrast, Theorem 2 is an explicit condition for $d-1$ Hamming distance with constant GC-content to produce the DNA coding sets which satisfy both constraints.

Theorem 1. *A code with length n can be smaller than a code with length* $n-1$ *with a minimum Hamming distance of* $0 \leq d \leq n$ *and* $0 < \omega < n$.

$$A_4^{GC}(n,\ d,\ \omega) \leq \left\lfloor \frac{2n}{\omega} A_4^{GC}(n-1, d, \omega-1) \right\rfloor, \tag{13}$$

$$A_4^{GC}(n,\ d,\ \omega) \leq\ \left\lfloor \frac{2n}{n-\omega} A_4^{GC}(n-1, d, \omega) \right\rfloor. \tag{14}$$

Proof. In the case of Equation (13), the sequence α_i for α_1 words with length n, $d_H(\alpha,\ \beta)$, and GC-content ω, there will be position j where $\lceil \omega\alpha_1/2n \rceil$ codewords have C nucleotide, or, at some position, it will be G. Otherwise, the average GC-content can be less than ω. Considering those codewords and deleting position j can generate $n-1$ and GC-content $\omega-1$ codes with minimum $d_H(\alpha,\ \beta)$. In contrast, Equation (14) is analogous, which only differs with GC-content for some position where $\lceil (n-\omega)\,\alpha_1/2n \rceil$ generates A's or T's. □

The inequalities in Equations (13) and (14) are applied to achieve the upper bounds on $A_4^{GC}(n,\ d,\ \omega)$ with $= d$, $n = \omega$, or $\omega = 0$ conditions. Similarly, different bounds can also be obtained by varying different orders; for instance, at constant $n = d$, Equation (13) can still be used after $n = \omega$ and Equation (14) after $\omega = 0$.

Theorem 2. *For the maximum code length n with minimum distance $d-1$ for the GC-content ω in lower bounds,*

$$A_4^{GC}(n,\ d,\ \omega) \geq \frac{\begin{pmatrix} n \\ \omega \end{pmatrix} 2^n}{\sum_{r=0}^{d-1} \sum_{i=0}^{\min\{\lfloor \frac{r}{2} \rfloor,\ \omega,\ n-\omega\}} \begin{pmatrix} \omega \\ i \end{pmatrix} \begin{pmatrix} n-\omega \\ i \end{pmatrix} \begin{pmatrix} n-2i \\ r-2i \end{pmatrix} 2^{2i}}. \quad (15)$$

Proof. By Equation (15), the numerator $\begin{pmatrix} n \\ \omega \end{pmatrix} 2^n$ provides the total codewords with GC-content ω. The denominator yields the codewords with $d-1$ distance for a sequence α, while $\begin{pmatrix} \omega \\ i \end{pmatrix} \begin{pmatrix} n-\omega \\ i \end{pmatrix} \begin{pmatrix} n-2i \\ r-2i \end{pmatrix} 2^{2i}$ denotes the lower bounds, which give the codewords of a sequence β with GC-content ω that must satisfy the $d_H(\alpha,\ \beta)$, for avoiding the error r. □

Apart from the GC-content, a reverse-complement constraint is integrated with this paper since we are employing the NJ algorithm for interaction energies $\varphi\left(\alpha_i,\ \beta_j\right)$ (Section 2) to control free energy for the secondary structures. To unfold the secondary structures before reading, let us consider a set of codewords, $\{AG,\ AC,\ TC,\ CA,\ TT\} \in \sum^n$. Any DNA code $\xi \subseteq \sum^n$ in DNA codebook with $2n$ length is constructed by defining a bijective map φ between the quinary alphabet Z_5 and $\sum^n$, and then the net code rate ($R = log_4 k/n$, k is number of DNA coding set, and n is number of sequence length) is, in this case, $log_4 5/2 \approx 0.58$ times of code. The bounds on free energy DNA code $\xi \subseteq \sum^n$ are presented in Proposition 2 to determine the secondary structure in a DNA sequence.

Proposition 2. *For all DNA sequences $\alpha_i = \alpha_1\alpha_2 \ldots \alpha_n$ and $\beta_i = \beta_1\ \beta_2 \ldots \beta_n$ in DNA code $\xi \subseteq \sum^n$, the free energy $E_{1,2n} \geq -2n$.*

From this proposition, the free energy $E_{i,j}$ reduces for DNA sequences over ξ. Hence, any DNA sequence α_i or β_i in $\xi \subseteq \sum^n$ will avoid secondary structures. In the above coding sets, with length $2n$, $E_{1,2n} \geq -5\left|\frac{2n}{2}\right| = -5n$. Now, from this proposition, we need to provide Theorems 3 and 4 to construct the model with which we avoid the secondary structure from any sequence.

Theorem 3. *Any DNA sequence (α_i or β_i) with length $2n$ in $\xi \subseteq \sum^n$ is free from the secondary structure if the stem length l* is more than 1 and minimum Hamming distance $d_H = d$.

Proof. Note that in any DNA sequence, if there is a secondary structure of stem length, then there will be 2 disjoint sub-sequences (α and β) with length l and there is $\alpha = \beta^{sc}$. The result will be contrapositive, such as if a DNA sequence frees from an SC (secondary-complement) sub-sequence with length l, then it will be freed from a secondary structure with a stem length of more than one. □

Theorem 4. *For any DNA code $A_4^{RC}(2n,\ K,\ d_H)$, the codeword over ξ, $\xi_{DNA} \cup\ \xi_{DNA}^c$ will be $(2n,\ 2K,\ d_H)$ if $d_H \leq n$, wherein $\xi_{DNA}^c = \{\alpha^c : \alpha \in \xi_{DNA}\}$.*

Proof. DNA code length and size follow the complement of DNA sequence in RC constraints for a given codeword over ξ, $\xi_{DNA} \cap \xi_{DNA}^c = \varnothing$. Similarly, note that $d_H(\alpha^c,\ \beta^c) = d_H(\alpha,\ \beta) \geq d_H$ and $d_H(\alpha^c,\ \beta) = d_H(\alpha,\ \beta^c) \geq n$. Hence, we have $\xi_{DNA} \cup \xi_{DNA}^c$ with minimum Hamming distance $\min\{d_H, n\} = d_H$ for the RC constraint. The results will follow the distance property of d_H. □

After constructing Propositions 1 and 2 for the upper and lower bounds' improvement and avoiding the secondary structure, respectively, we present the combinatorial constraints

by utilizing Proposition 3 [20]. This proposition leads to Theorems 5 and 6, improving the lower bounds of k-constraint length and avoiding particular errors from the number of sequences with errors r.

Proposition 3. *Suppose the bounds over* $A_4^{GC,RC}(n,\ d,\ \omega)$ *are concatenated with* $GC(\alpha,\ \beta)$ *and* $RC(\alpha,\ \beta)$ *constraints with the Hamming distance d for* $0 \le d \le n$ *and* $0 \le \omega \le n$. *We have 2 cases:*

If n is even,

$$A_4^{GC,RC}(n,\ d,\ \omega) = A_4^{GC,R}(n,\ d,\ \omega) \tag{16}$$

If n is odd,

$$A_4^{GC,R}(n,\ d+1,\ \omega) \le A_4^{GC,RC}(n,\ d,\ \omega) \le A_4^{GC,R}(n,\ d-1,\ \omega). \tag{17}$$

Proof. For any set of codewords with length n, if their complements in any subset replace all integers, the GC-content will be maintained due to the existence of a Hamming distance d between each codeword of α and β. However, the reverse or reverse-complement and Hamming distance between the codewords are not maintained generally. Subsequentially, if n is even, we can replace codeword α_i by its complements to generate a new codeword β_i with the first $n/2$ coordination, and then $H\left(\alpha_i, \beta_j^R\right) = H\left(\beta_i, \beta_j^{RC}\right)$ for all codewords α_i and β_j. In contrast, if n is odd, we can replace codeword α_i by its complements to generate a new codeword β_i with the first $(n-1)/2$ coordination, and then $\left|H\left(\alpha_i, \beta_j^R\right) - H\left(\beta_i, \beta_j^{RC}\right)\right| \le 1$ for all codewords α_i and β_j [20]. □

Theorem 5. *The code with combinatorial constraint is optimal for maximum code length n and minimum distance* $d = 2$ *for the GC-content* ω *in lower bounds if* $0 \le \omega \le n$ and $A_4^{GC,RC}(n,\ 2d,\ \omega) = \begin{pmatrix} n \\ \omega \end{pmatrix} 2^{n-2}$.

Proof. By Theorem 2, $A_4^{GC,RC}(n,\ 2d,\ \omega) \le \frac{1}{2} A_4^{GC}(n,\ 2d,\ \omega) = \frac{1}{2}\begin{pmatrix} n \\ \omega \end{pmatrix} 2^{n-1} = \begin{pmatrix} n \\ \omega \end{pmatrix} 2^{n-2}$. Similarly, by Proposition 3 (16), $A_4^{GC,RC}(n,\ 2d,\ \omega) \le \frac{1}{2} A_4^{GC,R}(n,\ 2d,\ \omega)$, and Theorem 4.5 of [38], $A_2^R(n,\ 2\ ) = 2^{n-2}$. In this argument, the set of all binary words for 2^{n-1} does not have palindromes for odd Hamming weight M, while the reverse of odd word weight is still odd weight when n is even, so 2^{n-1} words are distributed into 2^{n-2} pairs $\{\alpha, \alpha^R\}$, wherein each word from each pair indicates that $A_2^R(n,\ 2) = 2^{n-2}$. Thus, the product lower bounds $A_4^{GC,R}(n,\ d,\ \omega) \ge A_2(n,\ 2,\ \omega).A_2^R(n,2) = \begin{pmatrix} n \\ \omega \end{pmatrix} 2^{n-2}$ for the Hamming distance between 2 separated words of odd weight M should be at least 2; then, the inequality determines the Halving bound, $A_2^R(n,\ 2) \le \frac{1}{2} A_2(n,\ 2\ ) = 2^{n-2}$. □

The lower bounds with deletion or substitution errors ε and with $d \ge 2$ are not tight enough to generate the DNA library for high-density data storage. We can improve the lower bounds of the maximum number of sequences without errors r by constructing the redundancy of explicit DNA codes with $r/2 \log M$ by considering Shannon's relationship [32] (Equation (2)). The purpose of Theorem 6 is to improve the lower bounds of sequences without errors r; for which, the lower bounds with deletion and substitution errors ε are considered with fixed numbers of errors.

Theorem 6. *Let M, k, r, and ε be positive integers with r and $fixed$ ε. Suppose that $K > 3logM + \varepsilon$.* Then the redundancy of improved lower bounds is

$$\lfloor r/2 \rfloor \log M + \lfloor r/2 \rfloor \varepsilon - O(1).$$

Proof. For a sequence $\alpha \in \Gamma_M^{L+2}$, we consider the sequence $\alpha_1\alpha_2\alpha_3 \ldots \alpha_M$ in the way of descending lexicographic order. Each sequence contains discrete code, so each sequence of length K_ε occurs at most 2^ε times. Hence, the number of equivalence classes m is exactly the number of odd weights M with $2K_\varepsilon$ runs. This number is known to be (see [39,40], page 360).

$$m = \sum_{j=0}^{2^{K_\varepsilon}} (-1)^j \begin{pmatrix} 2^{K_\varepsilon} \\ j \end{pmatrix} \begin{pmatrix} 2^{K_\varepsilon} + M - j(2^\varepsilon + 1) - 1 \\ 2^{K_\varepsilon} \end{pmatrix}.$$

This expression for m is inconvenient to work with, so we assign a lower bound on m. W.L.O.G., we consider that for $1 \le i \le m_1$, where $m_1 \le m$ and $m_1 < i \le m$ with weight M of discrete codes,

$$m_1 = \begin{pmatrix} 2^{K_\varepsilon} \\ M \end{pmatrix} \le \frac{\begin{pmatrix} 2^K \\ M \end{pmatrix}}{2^{\varepsilon\ M}}. \tag{18}$$

while the number of equivalence classes with repetitions is

$$m - m_1 \le \sum_{K=1}^{M-1} \begin{pmatrix} 2^{K_\varepsilon} \\ K \end{pmatrix} K^{M-k}$$

where, in this expression, $\begin{pmatrix} 2^{K_\varepsilon} \\ K \end{pmatrix}$ gives the number of choices of these discrete codes, and K^{M-K} counts the remaining $M - K$ sequences as repetitions of the K discrete ones. Since $L > 3logM + \varepsilon$, when $K \le M - 2$, we have

$$\frac{\begin{pmatrix} 2^{K_\varepsilon} \\ K \end{pmatrix} K^{M-K}}{\begin{pmatrix} 2^{K_\varepsilon} \\ K+1 \end{pmatrix} (K+1)^{M-K-1}} = \frac{(K+1)^2}{2^{K_\varepsilon} - K} \left(\frac{K}{K+1} \right)^{M-k} < 1.$$

It follows that $\begin{pmatrix} 2^{K_\varepsilon} \\ K \end{pmatrix} K^{M-K}$ is increasing in K; hence,

$$m - m_1 = \sum_{K=1}^{M-1} \begin{pmatrix} 2^{K_\varepsilon} \\ K \end{pmatrix} K^{M-k} \le \begin{pmatrix} 2^{K_\varepsilon} \\ M-1 \end{pmatrix} M^2. \tag{19}$$

The Equation (18) is larger than Equation (19) w.r.t. discrete codes in each given sequence k:

$$\begin{aligned} \frac{\begin{pmatrix} 2^K \\ M \end{pmatrix}}{2^{\varepsilon M}} \Big/ \left(\begin{pmatrix} 2^{K_\varepsilon} \\ M-1 \end{pmatrix} M^2 \right) &= \frac{\left(2^L - M+1\right)\left(2^K - M+2\right)\left(2^K - M+3\right) \ldots 2^K}{M\left(2^{K_\varepsilon} - M+2\right)\left(2^{K_\varepsilon} - M+3\right) \ldots 2^{K_\varepsilon}} \cdot \frac{1}{2^{\varepsilon M} M^2} \\ &\ge \frac{\left(2^{K_\varepsilon} - M+1\right)\left(2^{K_\varepsilon} - M+2\right)\left(2^{K_\varepsilon} - M+3\right) \ldots 2^{K_\varepsilon}}{M^3\left(2^{K_\varepsilon} - M+2\right)\left(2^{K_\varepsilon} - M+3\right) \ldots 2^{K_\varepsilon}} \\ &= \frac{\left(2^{K_\varepsilon} - M+1\right)}{M^3} \ge 2^{K_\varepsilon - 1 - 3\ \log M} \ge 1. \end{aligned}$$

Hence,

$$m \leq \frac{\binom{2^K}{M}}{2^{\varepsilon M-1}}.$$

Now, let $\check{\mathrm{S}}$ be an error-correcting code. According to the pigeonhole principle, the sequence size is least $\frac{|\check{\mathrm{S}}|}{m}$ for a class $\S$, which indicates $\S \triangleq \check{\mathrm{S}}$. Therefore,

$$\left|\S\right| \geq \frac{\left|\check{\mathrm{S}}\right|}{m} \geq \frac{\left|\check{\mathrm{S}}\right|}{\binom{2^K}{M}\Big/2^{\varepsilon M-1}}. \tag{20}$$

Let $\Sigma \triangleq \{0,1\}^{\varepsilon}$ and

$$\mathcal{L}_k(n) \triangleq \{(\alpha_1[K_\varepsilon+1,K],\alpha_2[K_\varepsilon+1,K],\ldots,\alpha_M[K_\varepsilon+1,K]) \in \Sigma^M \mid \{\alpha_1,\alpha_2,\alpha_3,\ldots,\ \alpha_M\} \in \S\}.$$

We noted that while $\{\alpha_1\alpha_2\alpha_3\ldots\alpha_M\} \in \S$ is a coding set, at this point, we use the lexicographic order to assign the indices, $(\alpha_1[K_\varepsilon+1,K]\alpha_2[K_\varepsilon+1,K]\ldots\alpha_M[K_\varepsilon+1,K]) \in \Sigma^M$.

We suppose that $\mathcal{L}_k(n) \subseteq \Sigma^M$ is a code of minimum Hamming distance at least d; otherwise, if there are two codewords in $\mathcal{L}_k(n)$ that have the Hamming distance at most $d+1$, then both concerned codewords in $\S$ can be confusable. Thus, by deleting the length of ε suffixes, the concerned codes will be different in $\mathcal{L}_k(n)$. Thus, by using the Hamming bound on $|\mathcal{L}_k(n)|$ which is same as $\left|\S\right|$, we have

$$|\S| \leq \frac{2^{\varepsilon M}}{\sum_{i=0}^{\lfloor r/2\rfloor}\binom{M}{i}(2^\varepsilon-1)^i}. \tag{21}$$

By combining the Equations (20) and (21), we have

$$\left|\check{\mathrm{S}}\right| \leq \frac{2\binom{2^K}{M}}{\sum_{i=0}^{\lfloor r/2\rfloor}\binom{M}{i}(2^\varepsilon-1)^i}.$$

Hence,

$$\begin{aligned}\log\binom{2^K}{M} - \log\left|\check{\mathrm{S}}\right| &\geq \log\left(\sum_{i=0}^{\lfloor r/2\rfloor}\binom{M}{i}(2^\varepsilon-1)^i\right)-1\\ &= \lfloor r/2\rfloor\ \log M + \lfloor r/2\rfloor\varepsilon - O(1).\ \square\end{aligned}$$

In this paper, biologically constrained quaternary codes are pondered to use DNA primers economically. The pseudo-code of Algorithm 1 is utilized to generate the DNA library $\mathcal{L}_k(n)$ which is based on the optimal DNA codes Φ_{DNA} designed by a neural network. This algorithm produces the codes that satisfy the GC-content ω, reverse constraint, and Hamming distance $d_H(\alpha,\ \beta)$ using the quaternary encoding.

Algorithm 1. Proposed algorithm to construct DNA library $\mathcal{L}_k(n)$.

Input:
Premiere DNA codes $\xi \subseteq \sum^n$, optimal DNA codes Φ_{DNA}, GC-content ω, code length n, Hamming distance $d_H(\alpha, \beta)$, and reverse constraint;
Output:
DNA library $\mathcal{L}_k(n)$.

1. Convert binary data into $\xi \subseteq \sum^n$ by quaternary encoding $(4 \times 3^{n-1})$
2. Initiate the NN with activation gates $y^{in_j}(t)$ using Equation (6) and $y^{out_j}(t)$ using Equation (7) to encode the primers;
3. Generate optimal DNA codes Φ_{DNA} by output activation $y^{c_j^v}(t)$ using Equation (11) and LSTM layers;
4. Remove the codewords from Φ_{DNA} if that does not follow the GC-content ω (Proposition 1 and Theorem 2);
5. Reverse the DNA codes that enable secondary structures (Theorems 3 and 4) and avoid the codes that do not satisfy $d_H(\alpha, \beta)$ $(d-1)$;
6. Concatenate the bio-constraints $A_4^{GC,RC}(n, d, \omega)$ for n code length by Proposition 3 and Theorem 5;
7. Construct the error-correcting codes to produce the final DNA library $\mathcal{L}_k(n)$ by Theorem 6.

return: DNA library $\mathcal{L}_k(n)$ for DNA data storage.

5. Result Evaluations

This section elaborates on the improved lower bounds and DNA coding sets obtained by the proposed model of NN and combinatorial bio-constraints. Figure 2 illustrates a random sample of forward and reverse primers for the optimal DNA codes received after the NN implementation. These random DNA sequences were programmed in the Magma program [41] with different sequence lengths and a minimum Hamming distance. The aforementioned propositions and theorems were considered for program construction. As a result, we received *.cod* files with different lower bounds of DNA codes, satisfying the combinatorial bio-constraints for particular n and d. The codes in the *.cod* files were calculated in the Tables format. In addition, Figure 3 illustrates the numerical analysis by considering the coding rate and storage density of lower bounds given in these tables. Figure 3's analyses were drawn by using the Prism program.

Figure 2. A sample of received primers for the optimal DNA codes.

Tables 1 and 2 present the lower bounds obtained by our model of GC-content ω and $d_H(\alpha, \beta)$ with the NN. In each row, the upper entries in Table 1 are directly taken from [10], while the upper entries in Table 2 belong to [17]. The lower entries are used to compare our outputs with existing studies. The superscript i represents improved lower bounds and d indicates the decreased lower bounds, while the rest of the other bounds have almost the same lower bounds as compared to [10] and [17], respectively.

Table 1. Comparison of our lower bounds with [10] for $A_4^{GC}(n,\ d,\ \omega)$.

n/d	d = 3	d = 4	d = 5	d = 6	d = 7	d = 8	d = 9	d = 10
4	11 **12** [i]							
5	17 21	7 7						
6	44 59	16 19	6 7					
7	110 **143** [i]	36 **52** [i]	11 **19** [i]	4 4				
8	289 303	86 **115** [i]	29 36	9 10	4 4			
9	662 **864** [i]	199 **291** [i]	59 61	15 **31**[i]	8 7 [d]	4 5		
10	1810 1973	525 **604** [i]	141 **171** [i]	43 **51** [i]	7 **21** [i]	5 6	4 4	
11	4320 **5764** [i]	1235 **1716** [i]	284 **401** [i]	82 **125** [i]	29 **41** [i]	9 **17** [i]	4 5	4 4
12	12,068 11,618 [d]	3326 **4986** [i]	662 617 [d]	190 **711** [i]	58 72	22 **29** [i]	8 **11** [i]	4 4
13	41,867 57,322	7578 8113	1432 **2564** [i]	1201 1391	123 368	39 **71** [i]	13 21	6 **8** [i]

In Table 1, the lower bounds are based on GC-content ω and $d_H(\alpha,\ \beta)$ by deriving Proposition 1 and Theorem 1, and they are compared with Table 1 of [10], which uses the $4 \leq n \leq 13$ and $3 \leq d \leq 10$ inequalities to construct the DNA codes. We have compared our proposed model's results with [10] by considering the GC-content ω and $d_H(\alpha,\ \beta)$ with NN. As a comparison, 51% of bounds are improved, 5% have decreased, and 44% are almost the same lower bounds as in [10].

Similarly, in Table 2, the lower bounds are based on RC constraint and $d_H(\alpha,\ \beta)$ by deriving Proposition 1 and Theorem 2 and are compared with the Table 7 of the study [17], which considers the $4 \leq n \leq 10$ and $3 \leq d \leq n$ inequalities to design the DNA codes. In comparison, 64% of bounds are improved in our work, while 11% have decreased and 25% are almost the same as the lower bounds of [17]. However, the limitations of these bounds can be further improved by constructing new theorems or modifying Proposition 1 by varying the values of n and ω.

Apart from the lower bound improvements for the given constraints, the coding rates ($R = \frac{1}{n} log_4 L$, n is the sequence length number, and L is the total number of lower bounds in a sequence) have also been improved in a shorter sequence $(n-1)$. For instance, ref. [10] reported $R = 0.3036$ when $n = 8$ and $d = 5$, while our work reports the same coding rate (0.3034) with a shorter sequence when $n = 7$ and $d = 5$. Similarly, ref. [17] obtained $R = 0.4881$ when $n = 6$ and $d = 3$; in contrast, this work receives this coding rate (0.4857) when $n = 5$ and $d = 3$. The reported improved lower bounds have a better influence on the DNA library $\mathcal{L}_k(n)$ generation, indicating the proposed model's effectiveness for DNA code construction.

Table 2. Comparison of our lower bounds with [17] for $A_4^{GC}(n, d, \omega)$.

n/d	d = 3	d = 4	d = 5	d = 6	d = 7	d = 8	d = 9
4	12 12						
5	20 **29** [i]	8 **14** [i]					
6	58 **63** [i]	24 **27** [i]	8 **12** [i]				
7	125 118 [d]	44 51	17 22	7 **10** [i]			
8	324 **334** [i]	106 **124** [i]	35 **41** [i]	14 **17** [i]	5 **8** [i]		
9	713 **921** [i]	223 237	64 **94** [i]	24 23 [d]	10 **14** [i]	5 **7** [i]	
10	1906 2010	555 **913** [i]	159 163	51 48 [d]	20 21	10 **12** [i]	4 4

Regardless of this improvement, the 95% confidence interval (CI) mean (Figure 3a) of received bounds presents a breakthrough coding for DNA data storage in DNA computing. The bigger the interval, the more significant the development of coding for DNA data storage. As the purpose of individual RC constraints is to avoid the secondary structures in the DNA sequences, the RC constraints are not concerned with generating the lower bounds separately. However, the studies [11,13,18,20] motivate the idea of integrating the RC constraint with GC-content and Hamming distance in an assembled format to design new DNA coding sets. Taking advantage of their work, we generalize the RC constraint with Proposition 3 and Theorems 5 and 6 to generate the new DNA codes.

Table 3 is the collection of lower bounds with combinatorial constraints. Each column has upper and lower entries; the former is taken from the Table 8 of the study [20], and the latter is attained by our proposed computational model. The bold entries indicate the outperformed bounds of our proposed model over [20]. Likewise, the coding rates are compared in Figure 3b for $n = 8$ of our lower bounds with $n = 9$ of [20]. Our model designs the codes with almost the same R with $n - 1$ sequences. In addition, the underlined entries indicate the best-known codes of this work that have nine bounds total. In [20], Tables 8 and 9 present the best-achieved bounds which satisfy the GC-content, Hamming distance, and RC constraints; we only compare our results with Table 8 due to its particular inequalities (i.e., $3 \leq d \leq 11$). As Tables 1 and 2 are also based on these inequalities, we focus on a particular inequality in this paper for all the results.

The new lower bounds delivered by this work are better than the prior work. For instance, for $n = 10$ and $d = 5$, the size of our DNA codes is 22% greater than that of [20]. In another scenario, if we consider all the sequences at $d = 6$, the new improved DNA codes are still 36% better than [20]. These significant improvements are based on our proposed computational model that integrates a neural network with combinatorial bio-constraints. In addition, the size of these DNA codes is still capable of increasing as that of the best-known codes for the highest storage density. The storage density with our DNA codes for $n = 9$ to $n = 12$ and $d = 3$ to $d = 8$ is given in Figure 3c. The high storage density is received in lower Hamming distance, which is also based on the DNA coding sets of each sequence length. For the given particular lower bounds in Figure 3c, the highest density of 4.41 is attained for $d = 3$.

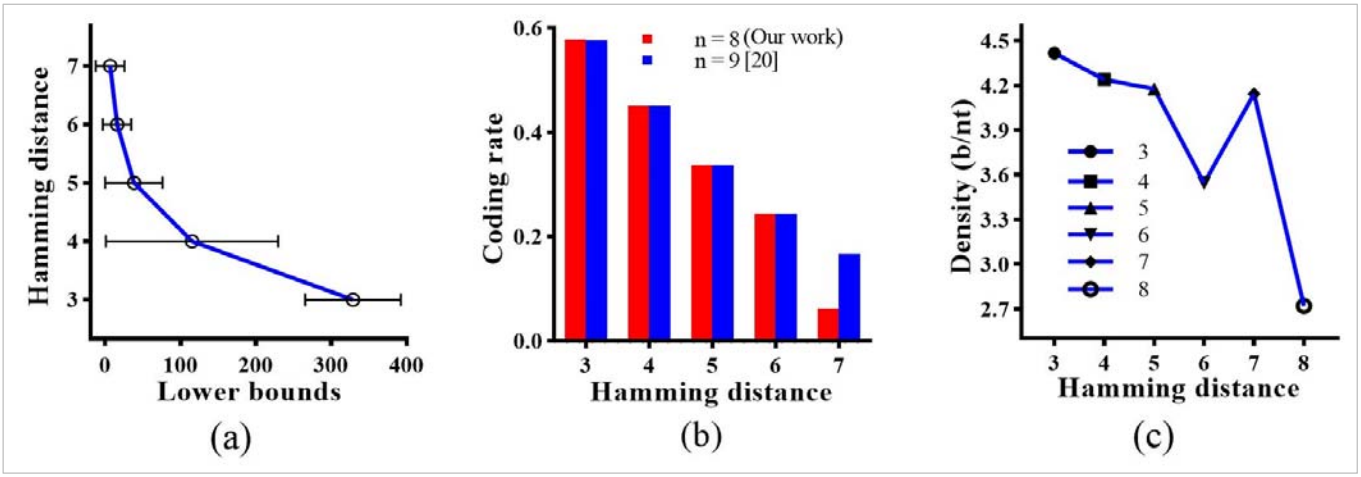

Figure 3. Lower bounds acquired by coding constraints with d_H: (**a**) The CI mean with lower and upper bounds of coding constraints with *GC* for $n = 8$. (**b**) The coding rate comparison between lower bounds is obtained by *RC* for our work ($n = 8$) and that of [20] $n = 9$. (**c**) The storage density with our DNA codes for $n = 9$ to $n = 12$ and $d = 3$ to $d = 8$.

Table 3. Comparison of our lower bounds with [20] for $A_4^{GC,RC}(n, d, \omega)$.

n/d	d = 3	d = 4	d = 5	d = 6	d = 7	d = 8	d = 9	d = 10
4	6 6							
5	15 **27**	3 4						
6	44 **67**	16 **21**	4 4					
7	135 **243**	36 **69**	11 **19**	2 2				
8	528 **617**	128 **148**	28 **42**	12 **15**	2 2			
9	1354 **1827**	275 **430**	67 **121**	21 **36**	8 11	2 2		
10	4542 **5914**	860 **1181**	210 **271**	54 **77**	17 **27**	8 8	2 2	
11	14,405 **23,713**	2457 **6429**	477 **961**	117 **557**	37 **59**	14 **23**	5 8	2 2
12	59,136 **67,761**	14,784 **19,132**	1848 **2062**	924 **1092**	87 **131**	29 **41**	12 **18**	4 6

Furthermore, the improvements of these lower bounds for any sequence length pioneer the DNA coding rates. A general analysis of Table 3 indicates that the same coding rate (R) is found in 73% of lower bounds with shorter sequences. For example, ref. [20] received $R = \frac{1}{12}log_4 87 = 0.2684$ when $n = 12$ and $d = 7$, while this work acquires the same coding rate (0.2673) when $n = 11$ and $d = 7$. Similarly, in another example, when $n = 10$ and $d = 4$, ref. [20] reported a 0.4874 coding rate; in contrast, this work delivers $R = 0.4860$ with the number of sequence lengths $n = 9$ at the same Hamming distance. In the case of best-known codes (bold underlined entries), our coding rate is better than [20], with a shorter sequence $(n - 1)$, i.e., $n = 8, d = 3$, and $R = 0.5652$, while this work reports $R = 0.5660$ when $n = 7$ and $d = 3$.

Thus, these analytical results present that the shorter sequences can achieve the same DNA storage density as the longer sequences. The improved lower bounds in various coding sets indicate the reduction in insertion and deletion errors in the DNA sequences, which enables the proposed computational model to avoid the non-specific hybridization process. In addition, Table 4 presents the DNA library $\mathcal{L}_k(n)$ satisfying the $A_4^{GC,RC}(n,\ d,\ \omega)$ constraints when $n = 10$ and $d = 7$, as in Table 3. The satisfaction of combinatorial constraints over the optimal DNA codes from the NN's output collaborated to improve the lower bounds of DNA coding sets, which emphasizes our proposed computational model.

Table 4. DNA coding sets for DNA library $\mathcal{L}_k(n)$ retrieved when $n = 10$ and $d = 7$.

GAGTCTAGAC	CTGTATGCAT	TACTAGACAG
GTCTGACATA	CACTACTGAC	ACTGTAGCAT
ATGACTCACT	GATACGACAT	CTACGTAGCA
TACTGTCACG	ACATCTGTCA	TGCACATGAC
AGCATACTCA	TACATCTGCT	GACATGACAG
CGATGTACTG	AGACGATGTC	TGTAGCTACA
CAGTAGATCA	TACGATCGAG	AGATCGACTG
GACTCATGAC	CACGTCTGAT	GCATAGTATC
ACTGACTACT	ACGCAGATAC	TGCGATACTA

6. Conclusions

An exciting research challenge in DNA data storage systems is to explore improved lower bounds by avoiding non-specific errors to generate high-density-based storage, which could store a large amount of information in a shorter sequence. In this paper, a novel computational model is offered to construct an extensive DNA library of oligonucleotides. It is accomplished by presenting a three-layer model that integrates a neural network (LSTM) and combinatorial bio-constraints, including GC-content, Hamming distance, and reverse-complement constraints. We derive the recursive expression in propositions and theorems to attain all possible large DNA coding sets by satisfying combinatorial constraints.

All DNA codewords in Tables 1 and 2 satisfy the GC-content and Hamming distance constraints and improve 51% and 64% of lower bounds compared to [10] and [17], respectively. The lower bounds presented in Table 3 are single error-correcting codes based on the concatenation constraints, while the underlined bounds exhibit the DNA sequences that have avoided secondary structures. Furthermore, the improvements in the lower bounds directly impact the coding rate. For example, results in Section 3 report that the shorter sequences can achieve the same DNA storage density as the longer sequences. It is concluded that the proposed computational model can store a large amount of data in a small number of DNA nucleotides that can improve the data density and reduce the DNA synthesis and sequence cost for a DNA-based data storage system.

In our results, there are still lower bounds that need to be improved by mutation strategies for high-density data storage. Similarly, the insertion and deletion errors can be further controlled by experimenting with the application-oriented bio-constraints, i.e., run-length constraints [42].

Author Contributions: Conceptualization, A.R., Q.J. and Y.W.; methodology, A.R.; software, A.R.; validation, A.R., Q.J. and Y.W.; formal analysis, Q.J., Y.W. and Q.Q.; investigation, Q.J., Y.W. and Q.Q.; resources, Q.J.; writing—original draft preparation, A.R.; writing—review and editing, A.R. and Y.W.; visualization, A.R.; supervision, Q.J.; project administration, Q.J.; funding acquisition, Q.J., Q.Q. and Y.W. All authors have read and agreed to the published version of the manuscript.

Funding: This work is supported by the National Key Research and Development Program of China under fund numbers 2021YFF1200100, 2021YFF1200104, and 2020YFA0909100.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: The data and codes used in this work are available at https://github.com/abdul-rasool/Coding-constraints-for-DNA-data-storage (accessed on 28 February 2021).

Acknowledgments: The authors would like to thank all the anonymous reviewers for their insightful comments and constructive suggestions that have obviously upgraded the quality of this manuscript.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Li, M.; Wu, J.; Dai, J.; Jiang, Q.; Qu, Q.; Huang, X.; Wang, Y. A self-contained and self-explanatory DNA storage system. *Sci. Rep.* **2021**, *11*, 18063. [CrossRef] [PubMed]
2. Yazdi, S.M.H.T.; Gabrys, R.; Milenkovic, O. Portable and Error-Free DNA-Based Data Storage. *Sci. Rep.* **2017**, *7*, 5011. [CrossRef] [PubMed]
3. Erlich, Y.; Zielinski, D. DNA Fountain enables a robust and efficient storage architecture. *Science* **2017**, *355*, 950–953. [CrossRef] [PubMed]
4. Blawat, M.; Gaedke, K.; Hütter, I.; Chen, X.-M.; Turczyk, B.; Inverso, S.; Pruitt, B.W.; Church, G.M. Forward Error Correction for DNA Data Storage. *Procedia Comput. Sci.* **2016**, *80*, 1011–1022. [CrossRef]
5. Grass, R.N.; Heckel, R.; Puddu, M.; Paunescu, D.; Stark, W.J. Robust Chemical Preservation of Digital Information on DNA in Silica with Error-Correcting Codes. *Angew. Chem. Int. Ed.* **2015**, *54*, 2552–2555. [CrossRef]
6. Goldman, N.; Bertone, P.; Chen, S.; Dessimoz, C.; LeProust, E.M.; Sipos, B.; Birney, E. Towards practical, high-capacity, low-maintenance information storage in synthesized DNA. *Nature* **2013**, *494*, 77–80. [CrossRef]
7. Church, G.M.; Gao, Y.; Kosuri, S. Next-Generation Digital Information Storage in DNA. *Science* **2012**, *337*, 1628. [CrossRef]
8. Yan, S.; Wong, K.-C. Future DNA computing device and accompanied tool stack: Towards high-throughput computation. *Future Gener. Comput. Syst.* **2021**, *117*, 111–124. [CrossRef]
9. Wang, Y.; Noor-A-Rahim, M.; Gunawan, E.; Guan, Y.L.; Poh, C.L. Construction of Bio-Constrained Code for DNA Data Storage. *IEEE Commun. Lett.* **2019**, *23*, 963–966. [CrossRef]
10. Limbachiya, D.; Gupta, M.K.; Aggarwal, V. Family of Constrained Codes for Archival DNA Data Storage. *IEEE Commun. Lett.* **2018**, *22*, 1972–1975. [CrossRef]
11. Benerjee, K.G.; Banerjee, A. On DNA Codes With Multiple Constraints. *IEEE Commun. Lett.* **2021**, *25*, 365–368. [CrossRef]
12. Rasool, A.; Qu, Q.; Jiang, Q.; Wang, Y. A Strategy-Based Optimization Algorithm to Design Codes for DNA Data Storage System. In *Algorithms and Architectures for Parallel Processing*; Springer International Publishing: Xiamen, China, 2022; pp. 284–299.
13. Chee, Y.M.; Ling, S. Improved lower bounds for constant GC-content DNA codes. *IEEE Trans. Inf. Theory* **2008**, *54*, 391–394. [CrossRef]
14. Zhang, J.X.; Yordanov, B.; Gaunt, A.; Wang, M.X.; Dai, P.; Chen, Y.J.; Zhang, K.; Fang, J.Z.; Dalchau, N.; Li, J.M.; et al. A deep learning model for predicting next-generation sequencing depth from DNA sequence. *Nat. Commun.* **2021**, *12*, 4387. [CrossRef]
15. Liu, Q.; Fang, L.; Yu, G.; Wang, D.; Xiao, C.-L.; Wang, K. Detection of DNA base modifications by deep recurrent neural network on Oxford Nanopore sequencing data. *Nat. Commun.* **2019**, *10*, 2449. [CrossRef]
16. Zhang, S.; Wu, J.; Huang, B.; Liu, Y. High-density information storage and random access scheme using synthetic DNA. *3 Biotech* **2021**, *11*, 328. [CrossRef]
17. Cao, B.; Li, X.; Zhang, X.; Wang, B.; Zhang, Q.; Wei, X. Designing Uncorrelated Address Constrain for DNA Storage by DMVO Algorithm. *IEEE/ACM Trans. Comput. Biol. Bioinform.* **2020**, 1. [CrossRef]
18. King, O.D. Bounds for DNA codes with constant GC-content. *Electron. J. Comb.* **2003**, *10*, R33. [CrossRef]
19. Milenkovic, O.; Kashyap, N. On the design of codes for DNA computing. In *Coding and Cryptography*; Ytrehus, O., Ed.; Springer: Berlin, Heidelberg, Germany, 2006; Volume 3969, pp. 100–119.
20. Aboluion, N.; Smith, D.H.; Perkins, S. Linear and nonlinear constructions of DNA codes with Hamming distance d, constant GC-content and a reverse-complement constraint. *Discret. Math.* **2012**, *312*, 1062–1075. [CrossRef]
21. Koumakis, L. Deep learning models in genomics; are we there yet? *Comput. Struct. Biotechnol. J.* **2020**, *18*, 1466–1473. [CrossRef]
22. Montana, D.J.; Davis, L. Training Feedforward Neural Networks Using Genetic Algorithms. In Proceedings of the Eleventh International Joint Conference on Artificial Intelligence, Detroit, MI, USA, 20–25 August 1989.
23. Hochreiter, S.; Schmidhuber, J. Long Short-Term Memory. *Neural Comput.* **1997**, *9*, 1735–1780. [CrossRef]
24. Muzammal, M.; Nasrulin, B. Renovating blockchain with distributed databases: An open source system. *Future Gener. Comput. Syst.* **2019**, *90*, 105–117. [CrossRef]
25. Jin, X.; Nie, R.; Zhou, D.; Yao, S.; Chen, Y.; Yu, J.; Wang, Q. A novel DNA sequence similarity calculation based on simplified pulse-coupled neural network and Huffman coding. *Phys. A Stat. Mech. Its Appl.* **2016**, *461*, 325–338. [CrossRef]
26. Deng, L.; Wu, H.; Liu, X.; Liu, H. DeepD2V: A Novel Deep Learning-Based Framework for Predicting Transcription Factor Binding Sites from Combined DNA Sequence. *Int. J. Mol. Sci.* **2021**, *22*, 5521. [CrossRef]
27. Song, W.; Cai, K.; Zhang, M.; Yuen, C. Codes with Run-Length and GC-Content Constraints for DNA-Based Data Storage. *IEEE Commun. Lett.* **2018**, *22*, 2004–2007. [CrossRef]
28. Siegel, P. Codes for Mass Data Storage Systems (Second Edition) (K. H. Schouhamer Immink; 2004) [Book review]. *IEEE Trans. Inf. Theory* **2006**, *52*, 5614–5616. [CrossRef]

29. Félix, B. On the embedding capacity of DNA strands under substitution, insertion, and deletion mutations. In Proceedings of the International Society for Optics and Photonics, San Jose, CA, USA, 17–21 January 2010.
30. Heckel, R.; Shomorony, I.; Ramchandran, K.; David, N. Fundamental limits of DNA storage systems. In Proceedings of the 2017 IEEE International Symposium on Information Theory (ISIT), Aachen, Germany, 25–30 June 2017; pp. 3130–3134.
31. Tulpan, D.; Smith, D.H.; Montemanni, R. Thermodynamic Post-Processing versus GC-Content Pre-Processing for DNA Codes Satisfying the Hamming Distance and Reverse-Complement Constraints. *IEEE-ACM Trans. Comput. Biol. Bioinform.* **2014**, *11*, 441–452. [CrossRef]
32. Shannon, C.E. A mathematical theory of communication. *Bell Syst. Tech. J.* **1948**, *27*, 379–423. [CrossRef]
33. Nussinov, R.; Jacobson, A.B. Fast algorithm for predicting the secondary structure of single-stranded rna. *Proc. Natl. Acad. Sci. USA* **1980**, *77*, 6309–6313. [CrossRef]
34. Peter Clote, R.B. *Computational Molecular Biology: An Introduction*; Wiley Series in Mathematical and Computational Biology; Wiley: Hoboken, NJ, USA, 2000.
35. Wu, Y.T.; Yuan, M.; Dong, S.P.; Lin, L.; Liu, Y.Q. Remaining useful life estimation of engineered systems using vanilla LSTM neural networks. *Neurocomputing* **2018**, *275*, 167–179. [CrossRef]
36. Rasool, A.; Jiang, Q.; Qu, Q.; Ji, C. WRS: A Novel Word-embedding Method for Real-time Sentiment with Integrated LSTM-CNN Model. In Proceedings of the 2021 IEEE International Conference on Real-time Computing and Robotics (RCAR), Xining, China, 15–19 July 2021; pp. 590–595.
37. Harding, S.E.; Channell, G.; Phillips-Jones, M.K. The discovery of hydrogen bonds in DNA and a re-evaluation of the 1948 Creeth two-chain model for its structure. *Biochem. Soc. Trans.* **2018**, *46*, 1171–1182. [CrossRef]
38. Marathe, A.; Condon, A.; Corn, R.M. On Combinatorial DNA Word Design. *J. Comput. Biol. A J. Comput. Mol. Cell Biol.* **2001**, *83*, 201–219. [CrossRef] [PubMed]
39. Charalambides, C.A. *Enumerative Combinatorics, CRC Press Series on Discrete Mathematics and Its Applications*; Chapman & Hall/CRC: Boca Raton, FL, USA, 2002.
40. Wei, H.; Schwartz, M. Improved Coding over Sets for DNA-Based Data Storage. *IEEE Trans. Inf. Theory* **2021**, *68*, 118–129. [CrossRef]
41. Cannon, J.; Bosma, W.; Fieker, C.; Steel, A.K. Handbook of Magma Functions. 2011. Available online: https://www.math.uzh.ch/sepp/magma-2.20.4-cr/HandbookVolume09 (accessed on 16 July 2021).
42. Paluncic, F.; Abdel-Ghaffar, K.A.S.; Ferreira, H.C.; Clarke, W.A. A Multiple Insertion/Deletion Correcting Code for Run-Length Limited Sequences. *IEEE Trans. Inf. Theory* **2012**, *58*, 1809–1824. [CrossRef]