Classical-to-Quantum Sequence Encoding in Genomics

Nouhaila Innan [1] Muhammad Al-Zafar Khan [2]

[1]Hassan II University of Casablanca [2]University of the Witwatersrand

Abstract

DNA sequencing allows for the determination of the genetic code of an organism and therefore is an indispensable tool that has applications in Medicine, Life Sciences, Evolutionary Biology, Food Sciences and Technology, and Agriculture. In this paper, several novel methods of performing Classical-to-Quantum data encoding inspired by various mathematical fields is presented, and their implementation is successfully exhibited within Bioinformatics. In particular, algorithms that draw inspiration from diverse fields such as Electrical and Electronic Engineering, Information Theory, and Neural Network architectures are introduced. The algorithms provided utilise Lossless Compression, and Information Entropy. Moreover, a contemporary method for testing encoded DNA sequences using Quantum Boltzmann Machines is proposed. This research contributes to developing Classical-to-Quantum data encoding methods in the science of Bioinformatics by introducing innovative algorithms that utilise diverse fields and advanced techniques. The findings offer insights into the potential of Quantum Computing in Bioinformatics, and have implications to drive future research in this area.

Introduction

Deoxyribonucleic Acid (DNA) is a molecule that contains the genetic instructions used in the development and functioning of all living organisms. Illustratively, it can be described as a long, double-stranded molecule made up of nucleotides, which are its composite building blocks. Each nucleotide in DNA consists of the sugar molecule deoxyribose [$C_5H_{10}O_4$], a phosphate group [-PO_4^{3-}], and a nitrogenous base: Either Adenine [$C_5H_5N_5$], Thymine [$C_5H_6N_2O_2$], Cytosine [$C_4H_5N_3O$], or Guanine [$C_5H_5N_5O$], denoted by **A**, **T**, **C**, and **G** respectively. The arrangement of these bases along the DNA molecule determines the *genetic code* or *sequence*, which is unique to every individual organism.

Quantum Computing (QC) exploits the Quantum Mechanical property of *superposition*, in which a state $|\psi\rangle$ can be written as a linear combination of the basis states $|0\rangle = \begin{pmatrix}1 & 0\end{pmatrix}^T$ and $|1\rangle = \begin{pmatrix}0 & 1\end{pmatrix}^T$, as $|\psi\rangle = \alpha|0\rangle + \beta|1\rangle$, where $\alpha, \beta \in \mathbb{C}$ are the complex probability amplitudes. Historically, QC has its roots in a 1982 proposition by the Nobel laureate Richard Feynman ($1918 - 1988$), who begged the question of whether a computer could harness the properties of quantum mechanics to simulate quantum systems (Feynman, 1982).

The fundamental unit of QC is the *qubit*, a portmanteau of the words "quantum" and "bit", which lives in a two-dimensional vector space called the *Bloch sphere*. A qubit is the quantum mechanical analogue of a classical bit, but instead of being in the "0" or "1" state exclusively, it can simultaneously be in the 0 and 1 states, with associated probabilities.

Below, the fundamental gate-based operations utilised in the subsequent algorithms presented are tabulated.

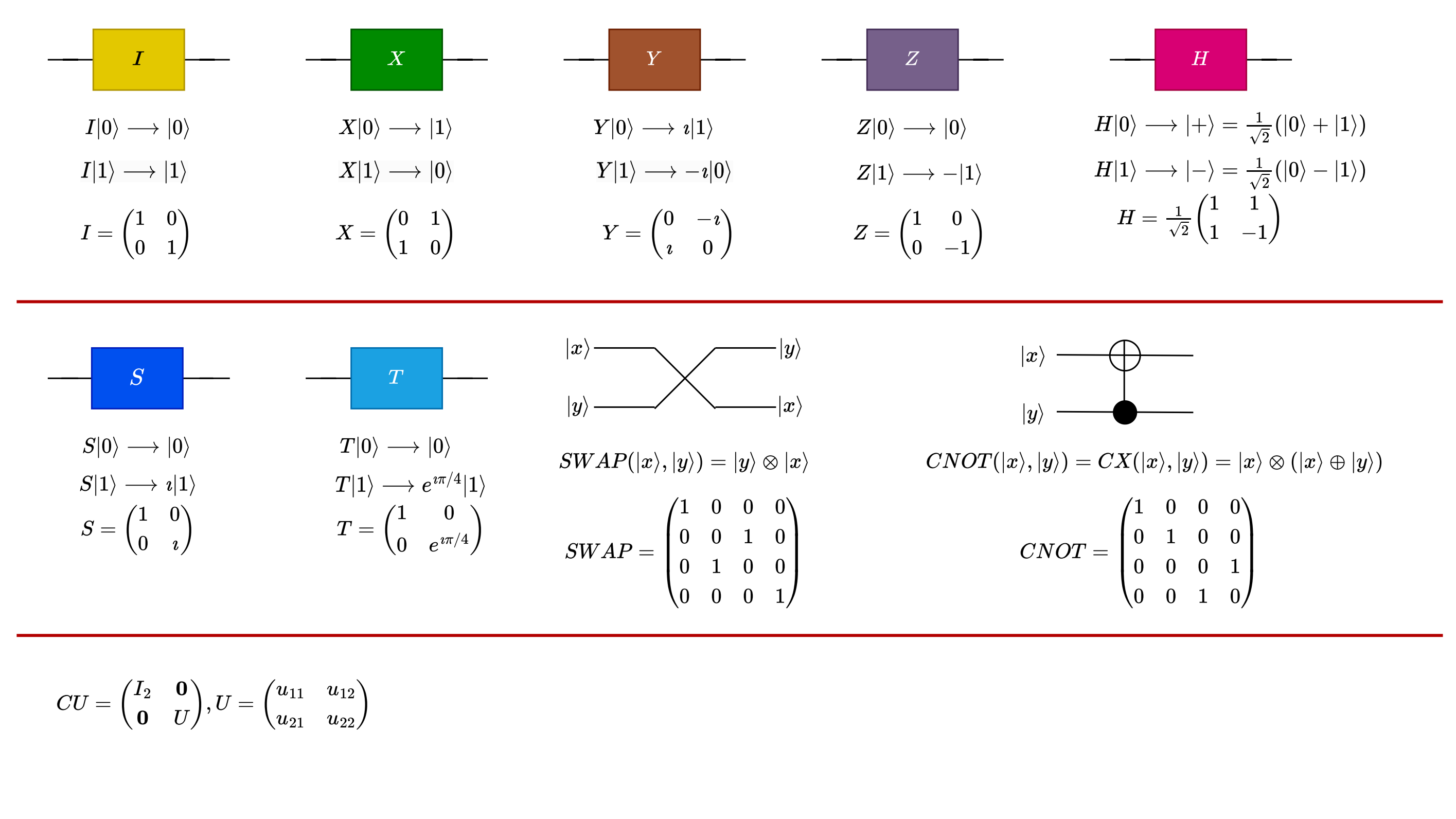

Figure 1. Single and two-qubit quantum gates.

In this paper, a deliberation on a class of new NISQ-era, a term coined by Preskill, 2018, algorithms inspired by a plethora of active fields of research spanning various STEM areas is presented.

Research Objectives

The present study investigates the following objectives:

- **Objective 1:** Development of novel Classical-to-Quantum encoding algorithms inspired by DNA sequencing.
- **Objective 2:** Development of algorithms for testing the viability of encoded DNA sequences.

Literature Review

Kathuria *et al*, 2020 presented two efficient inner product-based kernel classifiers for cataloguing individuals as "normal" or "having a disease" using functional genomic attributes as inputs. The classifiers presented used binary-valued features, which allowed for highly efficient data encoding and Hamming distance-based measurements. Using IBM's quantum computer, the algorithms were implemented. The classifiers required the same number of qubits for training samples and had low gate complexity after the states were prepared.

Boev *et al*, 2021 presented a method using quantum and quantum-inspired optimisation techniques to solve genome assembly tasks, with promising experimental results on both simulated data and the ϕX 174 bacteriophage. Their results suggested that the new generation of quantum annealing devices could outperform existing techniques for De Novo genome assembly, *videlicet* genome sequencing, where no reference sequences were available to refer to.

Sarkar *et al*, 2021 presented *QuASeR* as a reference-free DNA sequence reconstruction implementation via De Novo assembly on both gate-based and quantum annealing platforms. Furthermore, four implementation steps with a proof-of-concept example to target the genomics research were presented.

Nałęcz-Charkiewicz and Nowak, 2022 presented a proof for a De Novo assembly algorithm using a hybrid combination of CPU and the D-Wave quantum annealer QPU for calculations that were benchmarked against the results of a classical computer by using the Pearson correlation coefficient to detect overlaps between DNA readings. The training data consisted of synthetic and real data from a simulator, and this study was unique because actual organism genomes were sequenced. Due to the low number of qubits available on NISQ-era quantum computers. This study demonstrated that hybrid approaches are imperative in the field, and quantum annealers are a viable option for De Novo sequencing.

Algorithm 1: Lossless Compression

Huffman coding (Huffman, 1952) apportions varying-length codes to symbols based on their frequency of occurrence in the input data. At its core, Huffman coding represents frequently occurring symbols in the input data with shorter bit sequences and less frequently repeated symbols with longer bit sequences. In so doing, the algorithm achieves a higher compression ratio than fixed-length coding schemes that use the same number of bits to represent each symbol, regardless of its recurrence.

Architecturally, the algorithm builds a binary tree of nodes, where each leaf node represents a symbol, and its weight is equal to its frequency of occurrence in the input data. The algorithm starts by creating a list of all the symbols and their frequencies and then repeatedly combines the two least frequent symbols into a new node until all the symbols are contained within the tree. The bit sequence for each symbol is then determined by traversing the tree from the root to the corresponding leaf node and assigning a 0 or 1 bit for each left or right turn in the path. In this scheme, the number of qubits used in such a manner is optimised such that the encoded DNA sequence is lean.

```
Algorithm 1 QuantHuff($\mathcal{D}$)
  input a DNA sequence $\mathcal{D}$
  for each nitrogenous base $b$ in the DNA sequence $\mathcal{D}$ do do
      apply count($b$) $= C_b$ to tally up the frequency of each base
      apply rank_order($C_b$) $= R$ to rank the bases from lowest to highest
      apply build_tree($R$) $= \mathcal{T}$ to construct the root nodes of the Huffman tree
      apply cascade_sum($\mathcal{T}$) $= S$ to sum up the frequencies in pairs of two from the leftmost side and
  going up until the apex is reached
      apply gen_code($S$) $= G_b$ to assign binary digits along the edges of the tree
      if pos($E$) is left then
          code $\longleftarrow 0$
      else
          code $\longleftarrow 1$
      end if
      calculate the number of bits
                  $n_b = C_b \times \text{card}(G_b)$
      apply bin_encode($G_b$) $= |\phi_b\rangle$ to convert the bit strings to qubits
      calculate the encoded quantum state
                  $|\psi\rangle = \bigotimes_{\text{all bases}} |\phi_b\rangle$
  end for
  return encoded states $|\psi\rangle$, total bits required sum($n_b$)
```

In **Algorithm** 1 above, the code takes in a DNA sequence $\mathcal{D}$ as input, and for each nitrogenous base in the sequence, it counts the frequency of the base, ranks them according to frequency from lowest to highest from left to right. The code then graphically constructs the tree and then sums the frequency in pairs of two, starting from the leftmost side and building its way up until it reaches the tip of the tree. The algorithm then assigns binary bits (0 or 1) to the tree based on the following criteria: If the edge is to the left side, then it is allocated a 0, and if it is to the right side, it is allocated a 1.

Thereafter, the number of bits required to encode each nitrogenous base is calculated according to the simple formula of the product between the tally of each base in the sequence with the cardinality of each encoded bitstring of the respective base. Penultimately, the algorithm applies a simple binary encoding function that converts a 0 into a quantum state $|0\rangle$ and 1 into a quantum state $|1\rangle$. Lastly, the algorithm calculates the encode quantum state by taking the tensor product of each constituent base state. The algorithm then outputs the encoded quantum state and the sum of the number of bits to give the optimal number of bits required to encode the DNA sequence.

Consider, for example, the M13 filamentous bacteriophage universal reverse primer DNA sequence studied by Steffens *et al*, 1993: $\mathbf{5'}$ − CAGGAAACAGCTATGACC − $\mathbf{3'}$ commonly encountered in cloning and PCR testing. Below, the encoding scheme is applied, not accounting for the prefix and suffix numbers and hyphens, respectively, and provides a pictorial representation of the algorithm to obtain the encoded quantum state $|\psi\rangle$.

Table 1. Count of the nitrogenous bases in the M13 bacteriophage DNA sequence

Nitrogenous Base	Count
C	5
A	7
G	4
T	2
Total	**18**

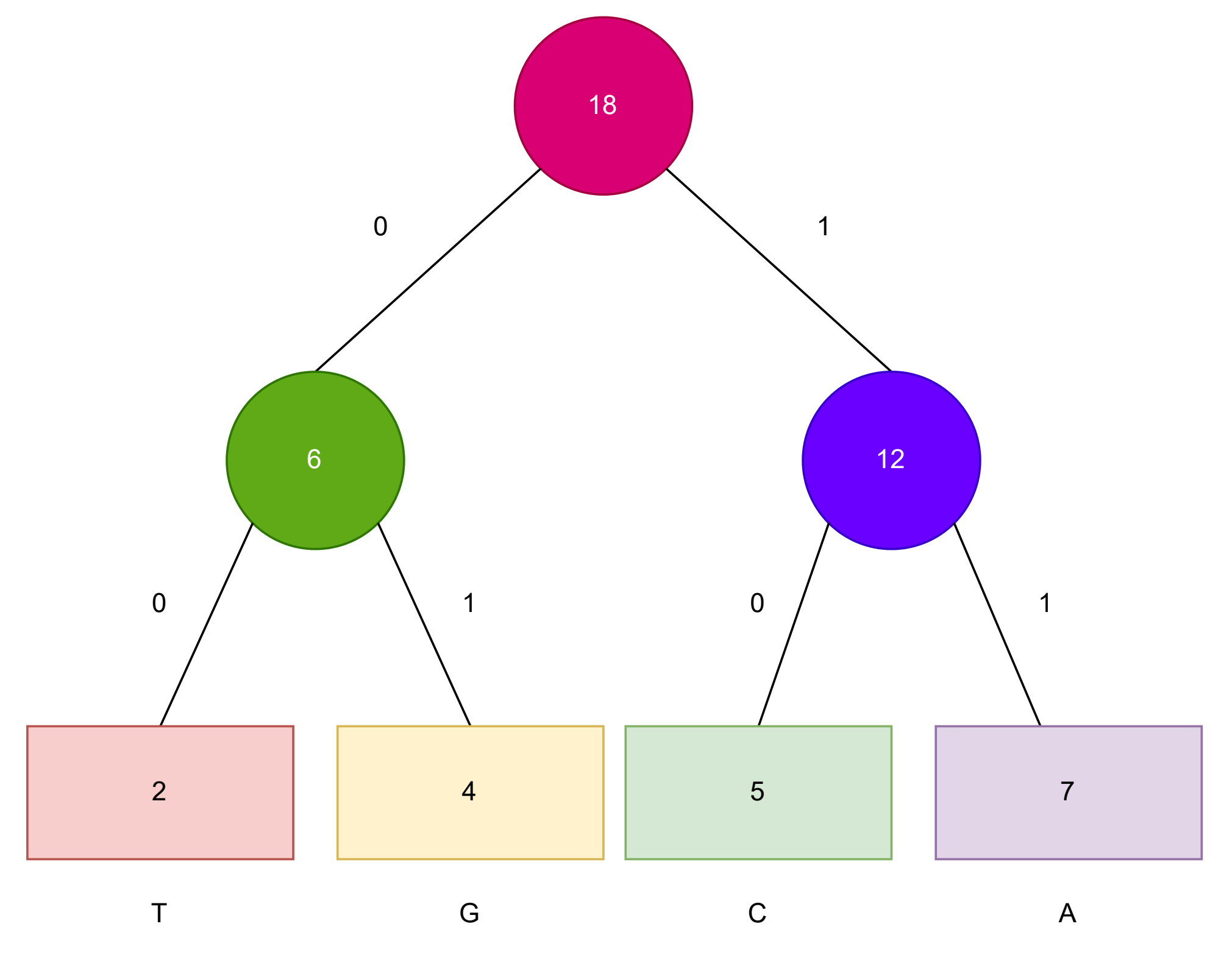

Figure 2. Graphical representation of the nitrogenous bases with their counts and codes along the edges respectively.

Table 2. Code determination for each nitrogenous base.

Nitrogenous Base	Count	Code	Bits
C	5	10	$5 \times 2 = 10$
A	7	11	$7 \times 2 = 15$
G	4	01	$4 \times 2 = 8$
T	2	00	$2 \times 2 = 4$
Total	**18**	-	**36**

Thus, binary encoding to convert the bits into qubits is applied, i.e. $0 \longrightarrow |0\rangle, 1 \longrightarrow |1\rangle$, to obtain the encoded quantum state for the M13 bacteriophage sequence:

$$\text{CAGGAAACAGCTATGACC} \longrightarrow |\psi\rangle = |10\rangle \otimes |11\rangle \otimes |01\rangle^{\otimes 2} \otimes |11\rangle^{\otimes 3} \otimes |10\rangle \otimes |11\rangle \otimes |01\rangle \otimes |10\rangle \otimes |00\rangle \otimes |11\rangle \otimes |00\rangle \otimes |01\rangle \otimes |11\rangle \otimes |10\rangle^{\otimes 2}. \tag{1}$$

Algorithm 2: Entropy-based Methods

In Information Theory, *entropy* is defined as the average amount of information contained in each message or symbol in a message. This average is calculated based on the probability distribution of the symbols in the message. The entropy of a system is the greatest when all possible messages are equally likely, and least when a message is certain to occur. Within ML, entropy is used to measure the impurity of a dataset. The information gain is calculated as the difference between the original dataset's entropy and the predicted values output from the model. ML algorithms based upon entropy methods try to maximise the information gain, leading to a more pure (less uncertain) dataset in the resulting groups.

Entropy-based methods for encoding are a natural consequence when comparing an encoded DNA sequence to a reference DNA sequence. It is also reasonable to assume that since DNA sequences have a high degree of inherent randomness.

Algorithm 2 $\texttt{SEncode}(\mathcal{D}, n)$

```
input a DNA sequence $\mathcal{D} \in \mathbb{S}$ of length $N$, $|\mathcal{D}| = N$ and the number of qubits $n$
initialise an $n$-qubit register in the $|0\rangle$ state, $|0\rangle^{\otimes n}$
for each nitrogenous base $b \in \mathcal{D}$ do
    calculate the probability of occurrence of the bases
        $p(b_i) = \frac{\text{Number of occurrences of } b_i}{N}, \quad i \in \{A, T, C, G\}$
    calculate the information entropy of the entire DNA sequence
        $H(\mathcal{D}) = -\sum_{i \in \mathcal{D}} p(b_i) \log_2 [p(b_i)]$
    calculate $K = \lceil \log_2(N) \rceil$, $M = \lceil \frac{N}{K} \rceil$
    apply fragment($\mathcal{D}$) to subdivide the DNA sequence into $M$ segments each of size $K$ such that $\mathcal{D} = \bigcup_{i=1}^{M} \mathcal{D}_i$
    for each fragmented DNA sequence do
        calculate the probability of the nitrogenous base in the segment
        calculate the entropy of each segment
        calculate the maximum entropy from all segments
            $H_{\max} = \max \{H(\mathcal{D}_1), H(\mathcal{D}_2), \ldots, H(\mathcal{D}_M)\}$
        normalise entropy in each segment
            $\widehat{H}(\mathcal{D}_i) = \frac{H(\mathcal{D}_i)}{H_{\max}}$
        apply the sort_low_high($\widehat{H}(\mathcal{D}_i)$) = $\mathscr{H}$ to order the normalised entropies from lowest to highest
        for each segmented DNA sequence $\mathcal{D}_i$ corresponding to $\mathscr{H}$ do
            synthesise quantum states $|\psi_i\rangle$ corresponding to each DNA segment
            apply a unitary operator to map $|0\rangle \longrightarrow |i\rangle$
                $U_i : |0\rangle \longrightarrow |i\rangle \otimes |0\rangle$
            apply a Hadamard gate to each qubit in the first register to create a superposition
        end for
        form the encoded state
            $|\psi\rangle = \bigotimes_{i=1}^{M} |\psi_i\rangle$
    end for
end for
return encoded states $|\psi\rangle$
```

Algorithm 2 takes a DNA sequence $\mathcal{D}$ of length N and the number of qubits n as input and produces an encoded quantum state $|\psi\rangle$ as output. The encoding is done by mapping each segment of the DNA sequence into a corresponding quantum state. The `SEncode` algorithm subdivides the DNA sequence into M segments, each of size K. The algorithm then calculates the probability of occurrence of each nitrogenous base in the DNA sequence, the entire DNA sequence's information entropy, and each segment's entropy. The maximum entropy among all the segments is then used to normalise the entropy in each segment. The normalised entropies are then sorted from lowest to highest.

The algorithm synthesises a corresponding quantum state for each segment and applies a unitary operator to map $|0\rangle$ to the corresponding segment index. A Hadamard gate is applied to each qubit in the first register to create a superposition. Finally, the algorithm forms the encoded quantum state by taking the tensor product of all the synthesised quantum states.

Illustratively, consider the DNA sequence $\mathcal{D}$ = ATCG, the number of nitrogenous bases $N = 4$, it is desired to encode the DNA sequence into a 2-qubit quantum state. The algorithm would first subdivide the DNA sequence into two segments of size 2: $\mathcal{D}_1$ = AT and $\mathcal{D}_2$ = CG. The algorithm would then calculate the probability of occurrence of each nitrogenous base, the information entropy of the entire DNA sequence, and the entropy of each segment.

As an example, assume that the normalised entropy of $\mathcal{D}_1$ is 0.5, and the normalised entropy of $\mathcal{D}_2$ is 0.5. The algorithm would synthesise quantum states corresponding to each segment by applying a unitary operator to map $|0\rangle$ to the corresponding segment index and applying a Hadamard gate to each qubit in the first register to create a superposition; the quantum states $|\psi_1\rangle = \frac{1}{\sqrt{2}}(|0\rangle + |1\rangle)$ and $|\psi_2\rangle = \frac{1}{\sqrt{2}}(|0\rangle - |1\rangle)$ is attained, corresponding to the segments $\mathcal{D}_1$ = AT and $\mathcal{D}_2$ = CG, respectively. Finally, the algorithm would form the encoded quantum state as $|\psi\rangle = |\psi_1\rangle \otimes |\psi_2\rangle = \frac{1}{2}(|00\rangle + |01\rangle - |10\rangle - |11\rangle)$.

Algorithm 3: Entropy Divergence Methods

Within the context of Genomics, it is noted that the in-vogue generalised entropy measure, the Kullback-Liebler divergence, has several *faux pas*. Namely:

1. It does not account for overlaps between the DNA and DNA reference sequences probability distributions.
2. It is not symmetrical. This means that the order of the respective probability distributions matter and, thus, significantly impact the results.
3. Since the DNA sequence is mapped to the computational basis $\{|0\rangle, |1\rangle\}$, it makes sense to have a divergence measure that is in the range $[0, 1]$. However, the KL metric is $[0, \infty)$.

It is proposed that Algorithm 3 which uses the Bhattacharyya divergence measure (Bhattacharyya, 1943; Bhattacharyya, 1946) to overcome the drawbacks of the KL divergence alluded to above.

Algorithm 3 $\texttt{NZ23}(\mathcal{D}, \mathcal{D}_{\text{ref}})$

```
input a DNA sequence $\mathcal{D}$ and a reference DNA sequence $\mathcal{D}_{\text{ref}}$ of the same length, $|\mathcal{D}| = |\mathcal{D}_{\text{ref}}| = N$
for each nitrogenous base $b \in \mathcal{D}, b_{\text{ref}} \in \mathcal{D}_{\text{ref}}$ do
    Calculate the probability of occurrence of the bases
        $P(b_i) = \frac{\text{Number of occurrences of } b_i}{N}, \quad i \in \{A, T, C, G\}$
        $Q(b_i^{\text{ref}}) = \frac{\text{Number of occurrences of } b_i^{\text{ref}}}{N}, \quad i \in \{A, T, C, G\}$
    Calculate the Bhattacharyya divergence between the $\mathcal{D}$ and $\mathcal{D}_{\text{ref}}$ probability distributions
        $D_B(P||Q) = -\log \left[ \sum_{\text{all bases}} \sqrt{P(b) \times Q(b)} \right]$
    Calculate the number of qubits required for some adjustable hyperparameter $0 \leq \alpha \leq 1$
        $n = \alpha \times \lceil D_B(P||Q) \rceil$
    Apply Amplitude Encoding to obtain state $|\psi\rangle$ based on $n$ qubits
end for
return encoded states $|\psi\rangle$
```

Suppose that we have a DNA sequence $\mathcal{D}$ of length $N = 10$: $\mathcal{D}$ = TACAGTTGCA, We also have a reference DNA sequence $\mathcal{D}_{\text{ref}}$ of the same length: $\mathcal{D}_{\text{ref}}$ = AGCTGACTCA The Bhattacharyya divergence is

$$D_B(P||Q) = -\log \sum_{b \,\in\, \{A,C,G,T\}} \sqrt{P(b) \times Q(b)} \approx 0.01.$$

The next step is determining the qubits needed to encode the DNA sequence. This is determined by a hyperparameter α between 0 and 1 and can be adjusted to trade-off between accuracy and computational resources. For this example, let's set $\alpha = 1$. Using the Bhattacharyya divergence and α, we calculate the number of qubits required as:

$$n = \alpha \times \lceil D_B(P||Q) \rceil = 1 \times \lceil 0.01 \rceil = 1.$$

This means we need one qubit to encode this DNA sequence into a quantum state. Finally, we obtain the quantum state using `Amplitude Encoding`. For this example, we can encode the state $|\psi\rangle$ as:

$$|\psi\rangle = \sqrt{P(\text{T})}\,|\text{T}\rangle + \sqrt{1 - P(\text{T})}\,|\neg\text{T}\rangle = \sqrt{0.3}\,|\text{T}\rangle + \sqrt{0.7}\,|\neg\text{T}\rangle,$$

where $|\neg\text{T}\rangle$ denotes a superposition of the other three bases (A, C, and G), and $\neg$ is the logical NOT operator.

Algorithm 4: Quantum Boltzmann Machine Methods

Quantum Boltzmann Machines (QBMs) (Amin *et al*, 2018) are a type of quantum neural network (QNN) architecture that has been proposed as a potential way to use the power of QC for ML tasks. They are inspired by classical Boltzmann machines which were proposed by Hinton and Sejnowski, 1985.

In a QBM, the binary inputs are replaced with qubits connected by a network of quantum gates, which are used to implement the probabilistic updates that drive the learning process.

The algorithm 4 is a proposed Quantum Machine Learning model for DNA sequence analysis. This algorithm encodes a given DNA sequence into quantum states, then trains using a QBM. The training involves updating the weights and biases of the machine using an optimisation algorithm, which minimises the cost function of the model.

Besides testing the proposed encoding algorithms, the `Qoltz` algorithm offers a new approach to analysing DNA sequences that combine the power of QC and ML.

Algorithm 4 Qoltz($\mathcal{D}$)

```
input a DNA sequence $\mathcal{D}$
initialise the quantum circuit with $n, \ell, m \in \mathbb{N}, 0 \leq \alpha \leq 1$, for the number of qubits, number of layers in the Boltzmann machine, number of steps in the optimisation algorithm, and the learning rate respectively with weights $\mathbf{W}$ and biases $\boldsymbol{\mathcal{B}}$
for each nitrogenous base $b$ in the DNA sequence $\mathcal{D}$ do
    apply bin_encode($b$) to encode the bases as binary numbers
    calculate the number of segments, $N$, to split the DNA sequence into, amongst $K$ varieties
    while card($\mathcal{D}$) $\neq 0$ do
        if card($\mathcal{D}$) $\equiv 0 \pmod 2$ then
            $N = \frac{\text{card}(\mathcal{D})}{K}$
        else
            $N = \left\lfloor \frac{\text{card}(\mathcal{D}) - 1}{K} \right\rfloor, \quad b_{\text{not sequenced}} = [\text{card}(\mathcal{D}) - 1] \pmod K$
        end if
    end while
    for each nitrogenous base $b$ in $N$ do
        apply quant_encode($s$) to map $0 \longrightarrow |0\rangle, 1 \longrightarrow |1\rangle$ to obtain the corresponding quantum states of each segment $|\psi\rangle$
        repeat
            split the encoded states into training and validation sets
            sample a mini-batch of encoded states
            calculate the energy for each quantum state
                $E(|\psi\rangle) = \sum_{\mathcal{L}=1}^{\ell} \sum_{\mathcal{N}=1}^{n} [w_0(\mathcal{L}, \mathcal{N}) |\psi_{\mathcal{N}}\rangle \otimes |\psi_{\mathcal{N}+1}\rangle + w_1(\mathcal{L}, \mathcal{N}) |\psi_{\mathcal{N}}\rangle \otimes |\psi_{\mathcal{N}+1}\rangle + w_2(\mathcal{L}, \mathcal{N}) |\psi_{\mathcal{N}}\rangle \otimes |\psi_{\mathcal{N}+1}\rangle] - \sum_{\mathcal{N}=1}^{n} \mathcal{B}_{\mathcal{N}} |\psi_{\mathcal{N}}\rangle$
            calculate the partition function
                $Z = \sum_{\text{all states}} \exp(-|\psi\rangle)$
            calculate the cost function, say
                $J = -\frac{1}{\text{card}(\mathcal{D})} \sum_{\text{all states}} \log \left[ \frac{1}{Z} \times \exp(-E(\psi)) \right]$
            apply an update rule for the weights for each state, say
                $w \longleftarrow w - \alpha \nabla_w J(W, |\psi\rangle)$
        until $J \longrightarrow 0$
    end for
end for
return encoded states $|\psi\rangle$
```

Using Boltzmann machines for Classical-to-Quantum data encoding, especially in DNA sequencing, is justifiable because the Boltzmann machine can learn pattern recognition and other structures in the input DNA sequence. Secondly, the Boltzmann machine can learn in an unstructured manner without requiring explicit labelling of nitrogenous bases and thus, proffers a promising approach to leveraging both classical and quantum computing simultaneously. QBMs are used to model the joint probability distribution of quantum data, such as the state of a quantum system or the outcomes of quantum measurements. By learning this distribution, they can be used for various QML tasks, such as Quantum Tomography, Quantum Generative Modelling, and Quantum Data Compression.

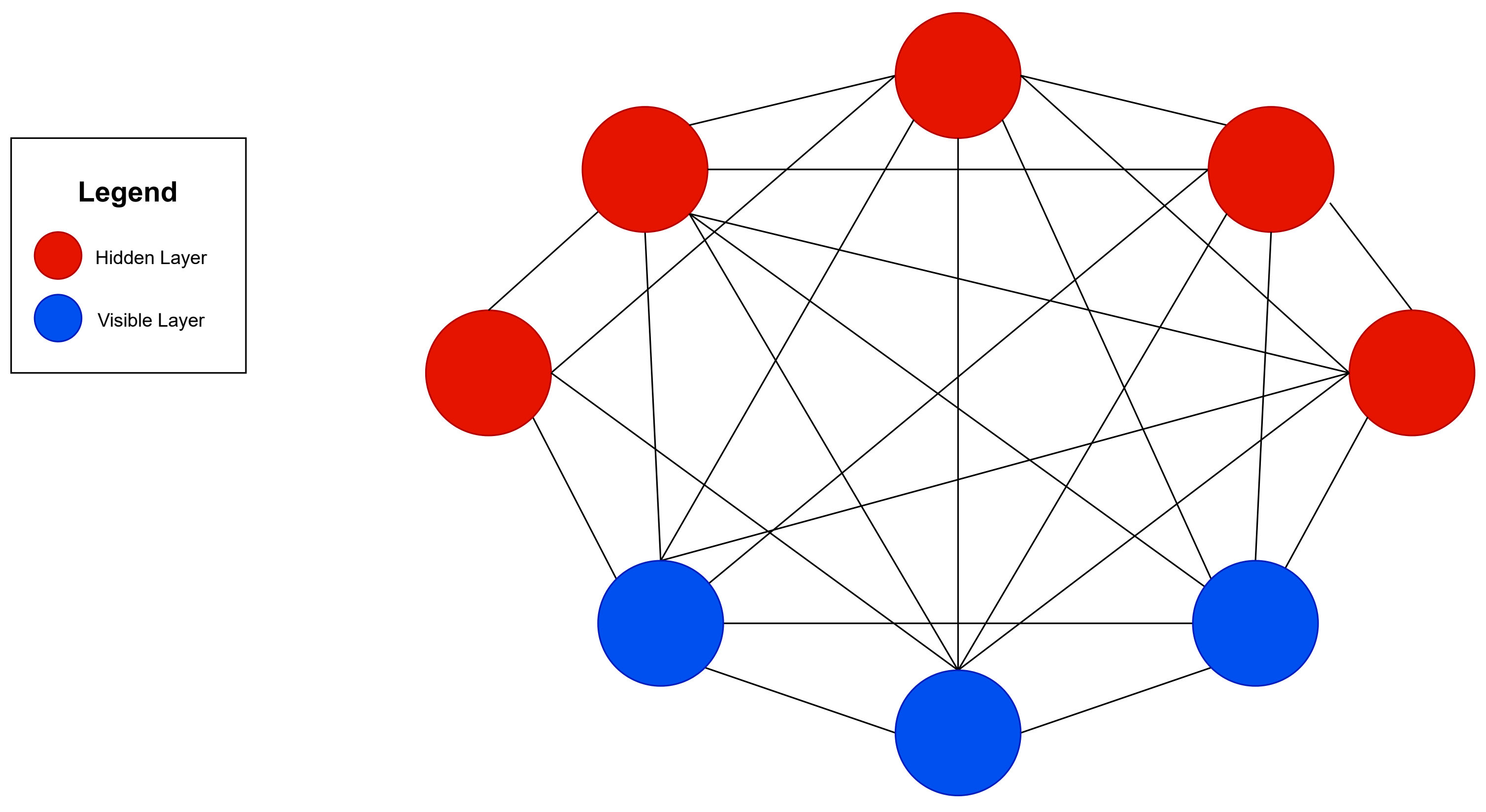

Figure 3. Architecture of a Boltzmann machine.

In Algorithm 4 above, the negative log-likelihood cost function has been used; however, the algorithm can be modified to include any cost function, such as MSE. In addition, we have used the SGD update rule; however, this can be changed to AdaGrad or ADAM. As an example, consider the DNA sequence **AAGT**; using the `Qoltz` algorithm (4) the DNA sequence is encoded into quantum states.
Firstly, the quantum circuit with the desired number of qubits, layers, steps, and learning rate, is initialised. Setting $n = 4$, $\ell = 2$, $m = 100$, and $\alpha = 0.01$, the weights and biases are also initialised. Thereafter, each nitrogenous base in the DNA sequence is encoded as binary numbers using the `bin_encode` function. The encoding for each base is as follows: A $\rightarrow 00$, C $\rightarrow 01$, G $\rightarrow 10$, T $\rightarrow 11$.
Therefore, the encoding for the DNA sequence **AAGT** is 0000111110; Next, the encoded string is split into segments, which implies that the string is split into $K = 2$ varieties. The length of the string is 10, which is divisible by 2. Thus, the string can be split into two equal segments: 00001111 and 10. Each segment is mapped to a quantum state using the `quant_encode` function, i.e. two qubits are used to encode each base and therefore, the first segment is encoded as $|\psi_1\rangle = |00\rangle^{\otimes 3} \otimes |01\rangle \otimes |11\rangle^{\otimes 3} \otimes |01\rangle$ and the second segment as $|\psi_2\rangle = |10\rangle$.
The quantum states can now be used as input to the QBM algorithm (4) to learn the underlying distribution of the DNA sequence. The algorithm will use the quantum states as input to a quantum circuit with n qubits, ℓ layers, and m optimisation steps. The algorithm will then use an update rule to adjust the weights and biases of the quantum circuit to minimise the cost function J. The resulting encoded states can be returned as the algorithm's output and tested using the same algorithm.

Results and Discussion

In this paper, three new algorithms for encoding Classical-to-Quantum data were presented, and practical examples of the usage and implementation of these algorithms were discussed. In addition, an algorithm inspired by QBMs is presented as a method of encoding, as well for checking the robustness in terms of information loss or gain, is presented.

For the first algorithm, `QuntHuff`, it is noted that the temporal computational complexity of the algorithm is $\mathcal{O}(k)$, where k is the number of steps required to implement the algorithm. We note that for the classical Huffman encoding scheme, the time complexity is $\mathcal{O}(k \log k)$, and it can be shown that since $\mathcal{O}(k \log k) > \mathcal{O}(k)$, the quantum algorithm proposed in this paper provided a speed up over its classical counterpart.

For the second algorithm, `SEncode`, it is noted that the algorithm proposed is mathematically convoluted with its requirements of calculating the entropy of segments. Furthermore, it has a time complexity of $\mathcal{O}(k^3)$ which makes it very expensive. Thus, it is advocated that this algorithm represents a step in the research direction of using entropy-based methods, and it is hoped that this will steer research initiatives in the direction of designing less mathematically elaborate, and less expensive algorithms.

For the third algorithm, `NZ23`, we advocate for the usage of the Bhattacharyya divergence. In particular, the algorithm proposed calculates the number of qubits required for encoding as well as the encoded state. The encoded state effectively makes use of the classical Amplitude encoding scheme: Given a classical datapoint $D = (x, y)$, amplitude encoding is performed as follows:

$$D \xrightarrow{\text{A.E.}} |\psi\rangle = (\mu + x)\,|0\rangle + (\nu + y)\,|1\rangle\,, \tag{2}$$

where $\mu, \nu \in \mathbb{C}$ are complex probability amplitudes. Additionally, the time complexity of the proposed algorithm is $\mathcal{O}(k)$, thus, this proves to be a viable algorithm that tells you both the number of qubits for encoding, as well as the encoded state.

For the fourth algorithm, `Qoltz`, it is noted that the algorithm contains many moving parts and a level of sophistication that requires advanced knowledge of Mathematics. However, it provides a unique method of encoding, as well as checking the information loss, or gain, in one holistic manner.

In the case of each algorithm, practical examples were provided. This demonstrates the effectiveness and viability of these techniques and can easily be implemented successfully in different contexts, and under different assumptions. Moreover, this helps in understanding the performance, strengths, and limitations of each algorithm under consideration.

Conclusion

It is noted that the algorithms presented in this paper form a small subset of the algorithm presented at the QuantumFormalism hackathon 2023 that required the development of novel algorithms, and resulted in a research paper currently undergoing peer review. Comprehensively, a summarised view of the algorithms is provided in the form of the graphical rendition below.

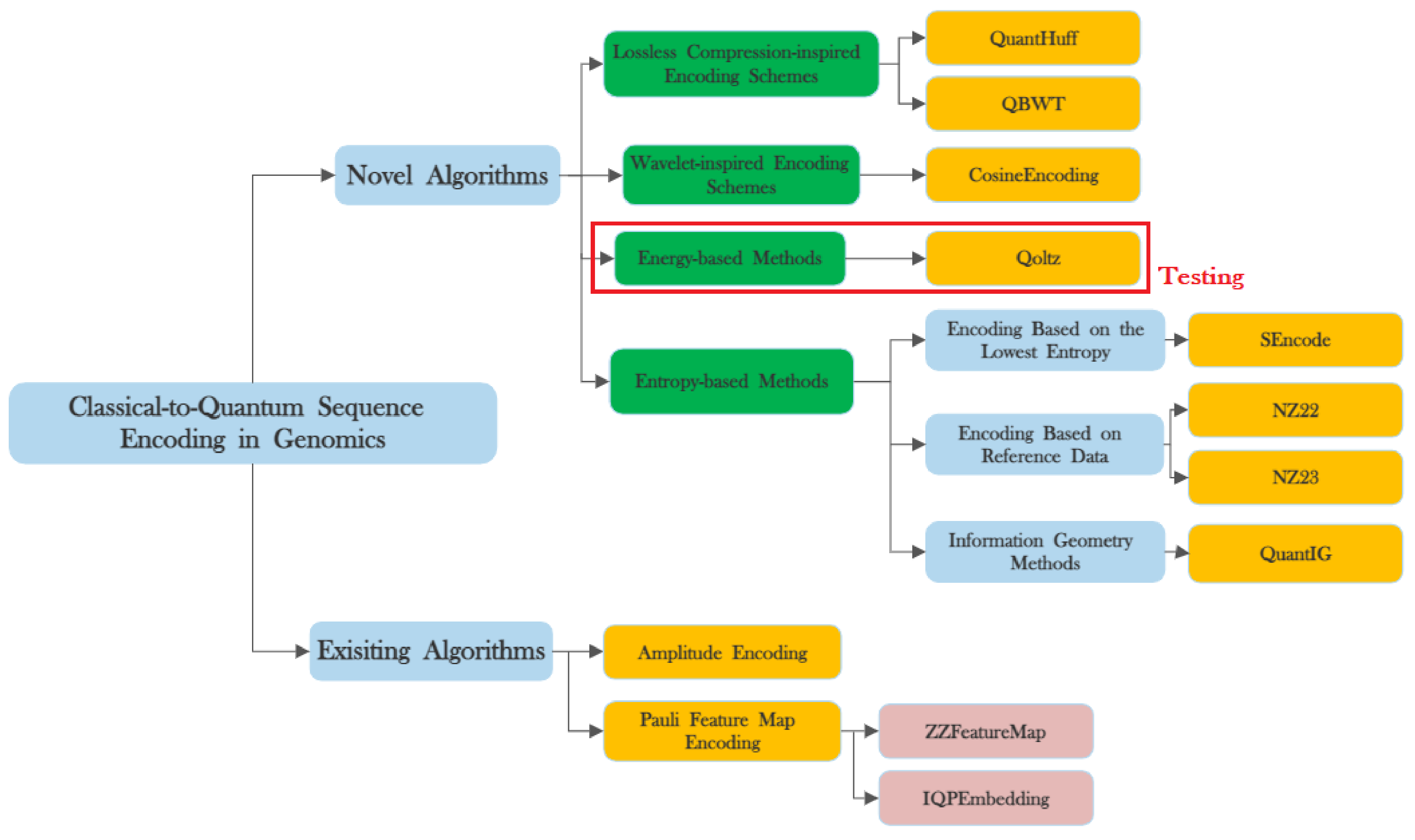
Classical-to-Quantum Sequence Encoding in Genomics
Novel Algorithms
Lossless Compression-inspired Encoding Schemes
QuantHuff
QBWT
Wavelet-inspired Encoding Schemes
CosineEncoding
Energy-based Methods
Qoltz
Testing
Entropy-based Methods
Encoding Based on the Lowest Entropy
SEncode
Encoding Based on Reference Data
NZ22
NZ23
Information Geometry Methods
QuantIG
Exisiting Algorithms
Amplitude Encoding
Pauli Feature Map Encoding
ZZFeatureMap
IQPEmbedding

Figure 4. Architecture of a Boltzmann machine.

In this paper, the quintessential NISQ-era algorithms that resulted from the Hackathon were presented. An important class of algorithms that have been omitted includes further lossless compression-based algorithms, namely the Quantum Burrows-Wheeler Transform, Wavelet-based methods, and those inspired by Computational Information Geometry, as can be seen in the tracing tree above. By no means has less importance been assigned to the excluded algorithms, however, those algorithms which have NISQ-era implementations, and can stir up further research initiatives, have been included.

Acknowledgements

Firstly, we would like to extend our sincere gratitude and appreciation to the QuantumFormalism community by the Zaiku Group for supporting the undertaking of this research project. Secondly, to Dr. Vesselin Gueorguiev, and Dr. Dimitri Papaioannou for their insightful comments and astute suggestions with regard to the structure and content of this paper.

References

[1] Feynman, R. P. "Simulating Physics with Computers". *Springer: International Journal of Theoretical Physics*, **21** (6/7), Berlin, Germany, pp. 467-488 (1982).

[2] Preskill, J. "Quantum Computing in the NISQ Era and Beyond". *Quantum*, **2**, Vienna, Austria, pp. 79-99 (2018).

[3] Kathuria, K., Ratan. A., McConnell, M., and Bekiranov, S. "Implementation of a Hamming Distance–like Genomic Quantum Classifier using Inner Products on ibmqx2 and ibmq_16_melbourne". *Springer: Quantum Machine Intelligence*, **2** (1). New York, United States of America, pp. 1-26 (2020).

[4] Boev, A. S., Rakitko, A. S., Usmanov, S. R., Kobzeva, A. N., Popov, I. V., Ilinksky, V. V., Kiktenko, E. O., and Federov, A. K. "Genome Assembly Using Quantum and Quantum–Inspired Annealing". *Nature*, **11** (1). New York, United States of America (2021).

[5] Sarkar, A., Al-Ars, Z., and Bertels, K. "QuASeR: Quantum Accelerated De Novo DNA Sequence Reconstruction." *PLoS One*, **16** (4). Cambridge, United Kingdom (2021).

[6] Nałęcz-Charkiewicz, K., and Nowak, R. M. "Algorithm for DNA Sequence Assembly by Quantum Annealing". *Springer: BMC Bioinformatics*, **23** (122), New York, United States of America (2022).

[7] Huffman, D. "A Method for the Construction of Minimum-Redundancy Codes". *Proceedings of the IRE*, **40** (9), United States of America, pp. 1098-1101 (1952).

[8] Steffens, D. L., Sutter, S. L., and Roemer, S. C. "An Alternate Universal Forward Primer for Improved Automated DNA Sequencing of M13." *BioTechniques: The International Journal of Life Science Methods*, **15** (4), London, United Kingdom, pp. 580-582 (1993).

[9] Bhattacharyya, A. "On a Measure of Divergence Between Two Statistical Populations Defined by their Probability Distributions". *Bulletin of the Calcutta Mathematical Society*, **35**, Calcutta, India, pp. 99-109 (1943).

[10] Bhattacharyya, A. "On a Measure of Divergence Between Two Multinomial Populations". *Sankhyā: The Indian Journal of Statistics*, **7** (4), Kolkata, India, pp. 401-406 (1946).

[11] Amin, M. H., Andriyash, E., Rolfe, J., Kulchytskyy, B., and Melko, R., "Quantum Boltzmann Machine". *American Physical Society: Physical Review X 021050*, **8** (2), Maryland, United States of America, pp. 1-11 (2018).

[12] Ackley, D. H., Hinton, G. E., and Sejnowski, T. J. "A Learning Algorithm for Boltzmann Machines". *John Wiley Sons: Cognitive Science*, **9** (1), New York, United States of America, pp. 147-169 (1985).