Chapter 4

RNA secondary structure estimation

4.1 Introduction

The four bases adenine (A), cytosine (C), guanine (G), and uracil (U) make up ribonucleic acid (RNA). Depending on its purpose, RNA can range in size from a few bases to millions of nucleotides. While messenger RNAs can be as long as 10^6 bases, transfer RNAs (tRNA) are among the shortest RNAs, ranging in size from 75 to 95 bases. The consecutive bases of an RNA primary sequence are connected by phosphate bonds. Unlike deoxyribonucleic acid (DNA), RNA is primarily single-stranded, but the bases of RNA tend to form base pairs with the help of hydrogen bonds that lead to the folding of RNA to attain a definite shape. The secondary structure of RNA refers to this folded form. The secondary structure allows RNA to carry out a number of crucial tasks. For instance, tRNA participates in translation by folding into the usual cloverleaf shape and the typical rho-independent hairpin structure contributes in transcription termination site in prokaryotes.

Figure 1-2 depicts the secondary structure associated with a hypothetical RNA sequence. Complementary base pairings A:U and G:C, as well as non-

complementary base pairs G:U, make up the three canonical base pairs in RNA. The space between two base pairs is sometimes referred to as a stacking region or stack, and the stem of secondary structure of RNA is formed by cascading stacking regions. A loop is formed by a series of unpaired bases between the stem region. A pseudoknot is created when two unpaired base sequences or loops inside a secondary structure join up to create an entangled tertiary structure. Pseudoknots play important roles in any biological system, including catalysis of RNase P ribozyme, RNA splicing, and/or recognition of tRNA-like structures [41].

Pseudoknots often have an important role in natural RNAs, specifically in viral RNAs [71]. Pseudoknots are also known to control the splicing, translation, ribosomal frame shifting [213][249], and gene expression [148] processes. About 40% of RNAs have pseudoknots. They also play an important role in RNA 3D folding [43]. H-type pseudoknots, in which the bases in the loop pair with bases in a single-stranded area that is not a part of the loop region, are the most prevalent type of pseudoknot. As illustrated in Figure 1-2, the other sorts of pseudoknots are HH-types, where the bases of one loop area pair with the bases of another loop region. There are also particular pseudoknots like the HL-type or LL-type, where L stands for an internal or bulging loop [67].

4.1.1 RNA secondary structure representation format

Different representations of RNA secondary structure are used for computational processing and visualisation. Dot-bracket notation, connection tables (CT), and BPSEQ are a few of the common ways to represent secondary structure.

4.1.1.1 Dot-bracket notation

A pair of parentheses is employed in this form to denote paired bases; the open-parenthesis "(" stands for the 5'-base and the matching close parenthesis ")" for the

3'-base. Unpaired bases are represented by the dot/period ".". Only straightforward pairs of matching parentheses are unable to clearly distinguish crossing base pairs when pseudoknots are present. As a result, a pseudoknotted secondary structure that permits crossing interactions is represented using an expanded dot-bracket notation with various sorts of brackets [254].

4.1.1.2 Connectivity Table (CT) format

In the CT format, the first row contains the length L of the RNA sequence. There are L subsequent rows, one per each base in the RNA sequence. Each row has six columns; the first column represents the index of a base starting from 1 and the second column represents the base itself. The third and fourth column represents the predecessor 5'-bases and successor 3'-bases index, respectively. The fifth column represents the index of the pairing base; the value is 0 for an unpaired base. The sixth or last column represents the index of base again. If the base is from the first RNA molecule, then the corresponding third column value is set to 0, and if the base is from the second molecule, then the corresponding fourth column value is set to 0 [254]. CT format is better than the BPSEQ format. CT format could be used to represent complexes consisting of two or more RNA molecules.

4.1.1.3 BPSEQ format

: In this format, secondary structure of RNA is represented in the form of a table. One base is shown in each row of the table. There are three columns in each row, with the first one specifying the base's index in the sequence, starting with 1 for the leftmost base. The base itself is specified in the second column. A 0 denotes that the pairing base indicated in the second column is unpaired. The third column displays the index of the related pairing base. The BPSEQ format is actually just a condensed version of the CT format [254].

4.1.2 RNA secondary structure visualizing tool

Proper visualization of RNA secondary structure motifs is essential to this research. Suitable visualization of the RNA structure motifs can help identify functional domains, compare secondary structures, and finding conserved regions across species. Some of the software available in the public domain to visualize RNA structure are FORNA [93], RNA secondary structure graphical rendering library [133], PseudoViewer [21], VARNA [34], and TraVeLer [40].

4.1.3 Computational methods used for RNA secondary structure prediction

Expensive and complex scientific experiments such as Nuclear Magnetic Resonance (NMR) spectroscopy and X-ray crystallography are often used to determine RNA secondary structure. However, it is useful to estimate RNA secondary structure with reasonable accuracy using interdisciplinary approaches based on computational algorithms. A generalized form of the objective function of these computational methods may be represented as follows:
Objective function (Z) : Determine a set of base pairs (Z)
Subject to:

1. A base pair (x:y) can be either of A:U, G:C or G:U
2. A base pair (x:y) is accepted if the difference between their positions in the sequence is ≥ 3
3. If a base pair (x:y) exists, then (x:z) doesnt exist

There are quite a few ways to determine RNA secondary structure that exist in literature which are briefly described in this section. There are broadly two ways to determine RNA secondary structure. First, using multiple homologous strains

of RNA or similar RNA sequences [74][14][231][17]. This method is reported as one of the widely accepted methods. Considering an alignment of a large number of homologous sequences, conserved base pairs in the secondary structure are estimated in these approaches. These methods are highly accurate, provided a large number of homologous sequences are available and the sequences are aligned with expert knowledge. The second approach is to estimate the secondary structure using only one sequence. Some of the notable proposals utilizing this approach are: dynamic programming approach based on a scoring system and free energy minimization [259][256][257], stochastic context free grammar approach which is based on probability of base pairs[96], genetic algorithm [222], backtracking of path matrix [106], thermodynamic RNA prediction[153] and learning based methods.

Another approach is finding the near maximum independent set (MIS) of chords of a circle graph, where the nucleotides are placed on the circumference of a circle graph. Base pairs are represented as the chords in the circle graph. MIS gives the largest number of vertices that are not adjacent to each other. In a real scenario, one base pairs with exactly one base, if any, and there would be no intersection of base pairs. So a planar circle graph with the maximum number of chords is supposed to provide a suitable RNA secondary structure. To determine MIS of a graph is known to be NP-complete[52]. Still, some methods have been proposed to determine MIS [53][79]. Parallel algorithm also been suggested in the literature to determine MIS [212][156][112]. This method is based on a single neuron model, which iterates over few hundred iterations to find the MIS. But some of the limitations of this method are, some parameters needs to be set at the start. These parameters need to be changed on every run, which would give new MIS, which further needs to be compared with previous runs and to keep the optimal one. If the number of bases in the RNA sequence is high enough, then one single run takes a large amount of time. The selection of parameters is also a concern, as the results do not follow any definite pattern, so on what interval should we increase or decrease the parameters are a question.

4.2 Improved method to predict RNA secondary structure based on MIS

In our approach to find MIS, we used python *igraph* package to identify all possible MIS on a single run. The algorithm used is explained in the method section 4.2.1.1 that considers the secondary structure having a large proportion of the base sequence coming under stem regions. Because of the limitation of the computing power of our server, we analyzed shorter RNA sequences and observed better results as compared to other methods.

4.2.1 Materials and Method

The more the stacking regions in RNA, the more stable the structure is, where the stacking region is the region between two base pairs. So, the requirement is to maximize the number of base pairs.

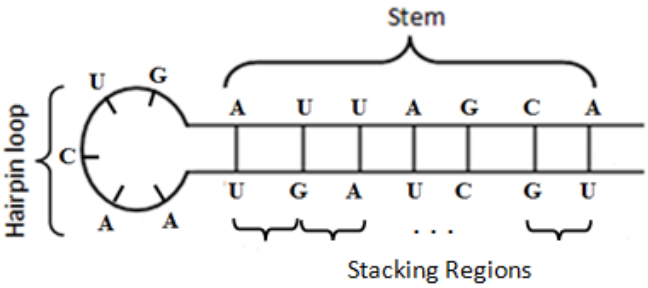

Figure 4-1: RNA Secondary Structure with Stem and Hairpin loop

To explain the method, say for a given RNA sequence 'AUCGC-CGGU', we find all possible base pairs using a base pairing matrix [216] as shown in Figure 4-2(i). The RNA sequence is taken as row and column header, for every possible base pair G:C, A:U, G:U, we mark 1 for the intersection of base pairs in the matrix. G:C, A:U are known as the Watson-Crick base pair, and G:U is known as non-Watson-Crick base pair. We take only the upper right triangular matrix for the subsequent steps, as the matrix generated is symmetric. Next, we consider a circle graph as shown in Figure 4-2(ii), with each nucleotides as it's vertex and

possible base pairs as chords. As stated in the literature, for a stable structure, the minimum number of nucleotides in the loop region is supposed to be at least 3 [216]. Taking this constraint, we remove certain base pairs, where two bases can pair, if there are more than two bases between them, as shown in Figure 4-2(iii). Then, we map the circle graph to an adjacency graph as in Figure 4-2(iv), we take all chords of the circle graph as new nodes of the adjacency graph and intersecting chords of the circle graph as new edges of the adjacency graph.

For a chord, say '2' between 'A' at node '1' and 'U' at node '9' as shown in Figure 4-2(iii), two variables are taken as '*from*' and '*to*', where '*from*' < '*to*', hence, *from* of chord '2' is '1' ($from(2) = 1$), similarly *to* of chord '2' is '9' ($to(2) = 9$) as the chord '2' emerges from vertex 1 and ends at 9.

For intersection of chords, we check the following conditions taking every two chords say 'a' and 'b' in circle graph as follows: $from(\mathrm{a}) < from(\mathrm{b}) < to(\mathrm{a}) < to(\mathrm{b})$, $from(\mathrm{b}) < to(\mathrm{a}) < to(\mathrm{b}) < to(\mathrm{a})$, $to(\mathrm{a}) = to(\mathrm{b})$, $to(\mathrm{a}) = from(\mathrm{b})$, $from(\mathrm{a}) = to(\mathrm{b})$, $from(\mathrm{a}) = from(\mathrm{b})$. The first two conditions check if two chords are intersecting. The last four checks if two chords have same vertex in common. For example, in Figure 4-2(iii) chord '2' and '11' intersect, as they have the same vertex '9' in common, so in the adjacency graph there will be an edge between '2' and '11'.

Next, we find all possible Maximum Independent Sets (MIS) of the adjacency graph, here in case we have only one MIS {2,5,7}, which are dark circled as shown in Figure 4-2(iv). In the next step, we choose the edges of MIS from the circle graph. So finally, we get a planar graph as shown in Figure 4-2(v) by selecting the chords of the circle graph named '2','5','7'.

In this example, we have only one MIS. But we may have multiple MIS for a given sequence. In that scenario, to resolve the conflict, first, we choose the structures with the maximum number of stacking regions. If the conflict still

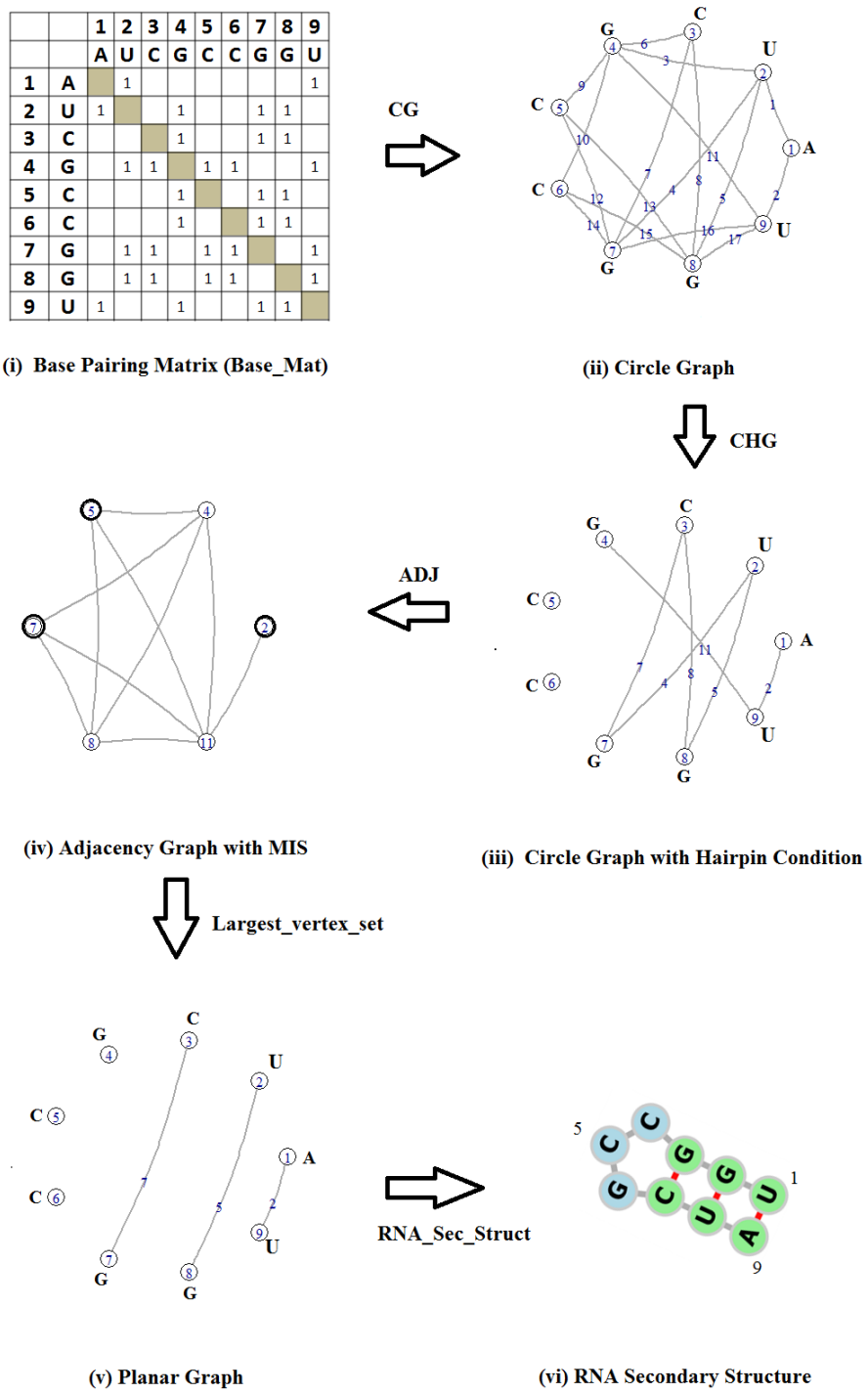

Figure 4-2: Steps followed to detect RNA Secondary Structure

exists, then we check for structures having maximum consecutive stacks. If conflict still persists, then we compare the energies of stacks based on the stacking energy table 4.1[221]. If the conflict still remains, we then compare the individual bond energies, along with loop energies also known as Tinocos stability number [216].

4.2.1.1 Algorithm RNA structure estimation

Following is the list of functions and variables/constants used in the algorithm

1. **BPM**(rna_seq): for a given RNA sequence, it returns all possible base pairs
2. **CG**(Base_Mat): Maps the Base_Mat to a circle graph, where:
   Bases in row header of Base_Mat = Vertices aligned to circumference of circle graph
   Base pairs of Base_Mat = Chords of circle graph, joining two bases
3. **CHG**(Cir_Graph): Returns a circle graph, by keeping only those base pairs where distance between bases is more than two, called hairpin condition
4. **ADJ**(Cir_Hpin_Graph): Maps Cir_Hpin_Graph to an adjacency graph, where:
   Chords of Cir_Hpin_Graph = Vertices of Adjacency Graph
   Intersecting chords of Cir_Hpin_Graph = Edges of Adjacency Graph
5. **Largest_vertex_set**(Adj_Graph): This function returns Maximum Independent set (MIS) of the adjacency graph, computed using python *igraph* package. It returns a 2D matrix MIS_Mat, containing all possible sets of MIS. MIS_Mat[i] represents each 1D matrix i.e. i^{th} row of MIS_Mat, where i ranges from 1 to no. of possible MIS.
6. **CS**(MIS_Mat[i]): A stack consists of two consecutive base pairs. This function returns an array containing the total number of stacks, in each MIS_Mat[i]

Table 4.1: Stacking Energy:- The table presents stacking energy, where the leftmost column represents the current base pair and the topmost row represents the next base pair in the stack. For example, value in row 2, column 1 represent the energy when C/G is followed by A/U [221].

	A/U	C/G	G/C	U/A	G/U	U/G
A/U	-0.9	-1.8	-2.3	-1.1	-1.1	-0.8
C/G	-1.7	-2.9	-3.4	-2.3	-2.1	-1.4
G/C	-2.1	-2.0	-2.9	-1.8	-1.9	-1.2
U/A	-0.9	-1.7	-2.1	-0.9	-1.0	-0.5
G/U	-0.5	-1.2	-1.4	-0.8	-0.4	-0.2
U/G	-1.0	-1.9	-2.1	-1.1	-1.5	-0.4

7. **Max(Count_Stack)**: It returns the maximum number of stacks comparing each MIS_Mat[i], and the count of how many maximum values

8. **CQS**(MIS_Mat[i]): This function returns an array containing total number of highest consecutive stacks in each MIS_Mat[i]

9. **Max(Count_Consqutv_Stack)**: It returns the maximum number of consecutive stacks comparing each MIS_Mat[i] and the count of how many maximum values

10. **SE**(MIS_Mat[i]): It returns the total Stacking energy as per the table given in Stacking Energy Table 4.1[221]

11. **Min(Stack_Energy)**: It returns the minimum stacking energies comparing each MIS_Mat[i]

12. **TSN**(MIS_Mat[i]): It returns Tinocos Stability Number comparing each MIS_Mat[i], for a particular MIS_Mat[i], TSN is the sum of Base Pair Energy (BP), Hairpin Energy (HP), Bulge loop energy (BL) and Interior Loop Energy (IL)

$$TSN = \sum BP + \sum HP + \sum BL + \sum IL \tag{4.1}$$

Table 4.2: Tinoco's Stability Number

Size	HP	BL	IL
< 2	NA	-2	NA
2	NA	NA	-4
3	-5	-3	-5
4 to 7	-6	NA	-6
> 7	-7		-7
4 to 15	NA	-5	NA
> 15	NA	-6	NA

The energies are as follows:

$$BP = \left\{ \begin{array}{l} 1, \text{ for A:U base pair} \\ 2, \text{ for G:C base pair} \\ 0, \text{ for G:U base pair} \end{array} \right\} \tag{4.2}$$

13. **Max(Tinoco_Stability_No)**: It returns the Maximum Stability Number when compared to each MIS_Mat[i].

14. **RNA_Sec_Struct**(MIS_Mat[i]): This function represents MIS_mat[i] as a dot-bracket notation, where brackets '(' or ')' represent nucleotides that participate in a base pair and '.' represents nucleotides that do not participate in base pair. The corresponding RNA secondary structure can be visualized from the dot-bracket notation.

```
Result: Estimated RNA secondary structure
Base_Mat = Compute BPM(rna_seq)
Cir_Graph = Compute CG(Base_Mat)
Cir_Hpin_Graph =Compute CHG(Cir_Graph)
Adj_Graph = Compute ADJ(Cir_Hpin_Graph)
MIS_Mat = Largest_vertex_set(Adj_Graph)
if length(MIS_Mat) = 1 then
    Compute RNA_Sec_Struct(MIS_Mat)
else
    Count_Stack = CS(MIS_Mat[i])
    Count_Stack_Max = Max(Count_Stack)
    if length(Count_Stack_Max) = 1 then
        Compute RNA_Sec_Struct(MIS_Mat[i])
    else
        Count_Consqutv_Stack = CQS (MIS_Mat[i])
        Count_Consqutv_Stack_Max = Max(Count_Consqutv_Stack)
        if length(Count_Consqutv_Stack_Max) = 1 then
            Compute RNA_Sec_Struct(MIS_Mat[i])
        else
            Stack_Energy = Compute SE(MIS_Mat[i])
            SE_Min = Min(Stack_Energy)
            if length(SE_Min) = 1 then
                Compute RNA_Sec_Struct(MIS_Mat[i])
            else
                Tinoco_Stability_No = Compute TSN(MIS_Mat[i])
                Tinoco_Stability_No_Max = Max(Tinoco_Stability_No)
                for all MIS_Mat[i] with Tinoco_Stability_No_Max
                Compute RNA_Sec_Struct(MIS_Mat[i])
            end
        end
    end
end
```

The implementation of the algorithm is available in a web portal from Tezpur University (TU web server), which is accessible at the link http://14.139.219.242:8003/rna_struct

4.2.1.2 Description of how to use the TU web server:

Step I: Enter nucleotide sequence of RNA: The first step is to provide nucleotide(base) sequence of RNA for which secondary structure is to be detected.

Step II: Enter the restrictions for each nucleotide (Optional): This step is optional, where the user is provided with an option to impose restriction on bases, of which to pair and which not to. For every base, a 'x' is to be entered to restrict the base to pair, and a '.' to allow the base to pair.

Step III: Select the base pairs to keep: In this step, we provide an option to the user; to select base pairs that are to be included in RNA structure detection.

Step IV: Enter e-mail id: This step is optional; an email-id can be provided; if the user wants the results in their email.

In the next step, the user may hit the Calculate button to view the results, a dot-bracket notation, a circle graph of RNA structure, and a link [93] to visualize the RNA structure is also provided.

4.2.1.3 Performance Measurement

To determine the accuracy of our method and other known methods as provided in web servers of Vienna RNA fold [116], RNAStructure [164], and Cofold [153] in comparison to original RNA structures, we perform sensitivity (SS), specificity (SP) and correlation coefficient measures as follows:

$$SS = \frac{TP}{TP + FN}, \quad SP = \frac{TP}{TP + FP}$$

	BP predicted: No	BP predicted: Yes
BP exists: No	True Negative (TN)	False Positive (FP)
BP exists: Yes	False Negative (FN)	True Positive (TP)

Table 4.3: **Comarative Result**. L: Sequence Length (number of bases), SS: Sensitivity, SP: Specificity and CC: Correlation coefficient.

Sequence Name	L	Vienna RNAfold web server			RNAStructure web server			Cofold web server			TU web server		
		SS	SP	CC	SS	SP	CC	SS	SP	CC	SS	SP	CC
1F1T	38	1.00	1.00	100.00	1.00	1.00	100.00	1.00	1.00	100.00	**1.00**	0.93	96.36
1RAW	36	0.75	0.75	75.00	0.75	0.75	75.00	0.75	0.75	75.00	**0.80**	0.67	73.03
2JTP	34	1.00	1.00	100.00	0.85	1.00	91.99	f1.00	1.00	100.00	**1.00**	1.00	**100.00**
3DKN	32	0.75	0.75	75.00	0.75	0.75	75.00	0.75	0.86	80.18	**1.00**	0.73	**85.28**
E_coli_16S_rRNA	38	0.77	0.77	76.92	0.77	0.77	76.92	0.92	1.00	96.08	**1.00**	0.87	**93.09**
R17_Viral_RNA	55	0.90	1.00	95.12	0.90	1.00	95.12	1.00	1.00	100.00	**1.00**	1.00	**100.00**

$$CC = \sqrt{\left(\frac{TP}{TP+FN}\right) * \left(\frac{TP}{TP+FP}\right)}$$

where the confusion matrix is provided below, BP means Base pair:

4.2.2 Results and Discussion

For this study, RNAs which has been used in the literature are taken for comparison analysis. The first four RNAs depicted in Table 4.3, have been collected from the online database (http://server3.lpm.org.ru/urs/struct.py) named Universe of RNA structures. In the table 4.3, the sequence IDs given are the PDBId of RNA sequence. These RNA structures are experimentally obtained either by NMR or X-ray crystallography and are considered original RNA secondary structures. We also extracted dot-bracket notation for each RNA structure, for the comparison of RNA structures from different computational sources [74][12][153]. The last two RNAs are taken from available research literatures [212][202][112].

SS is the probability of correctly predicting base pairs, whereas SP is the probability that a base pair prediction is correct [8]. From the above table, we can say that in terms of sensitivity (SS) our algorithm has a higher probability of

predicting correct base pairs as SS is almost 1.0 in almost all cases, as compared to other methods. Whereas the specificity (SP) measure is comparable of our web server to other methods in the sequences of 3DKN, 1RAW and 1F1T. In our method, the correlation coefficient measure performs better in the case of 2JTP, 3DKN, *E_coli*_16S_rRNA and R17_Viral_RNA, but other methods perform well in case of 1F1T, 1RAW.

In this study, we proposed a method that determines the maximum possible base pairs in an RNA secondary structure with no intersections. We used the Maximum Independent Set (MIS) based approach which gives all possible combinations of base pairs that are maximum in number. The computational time complexity of this approach is O(nmμ), where n is the number of vertices, m is the number of edges and μ is the number of maximum independent sets of the circle graph [218]. Our proposed method not only maximizes the number of base pairs but stacking regions also, as it is known that the more the stacks in a RNA secondary structure, the more stable the structure will be.

It has been seen that in small RNAs not having bifurcation, the maximum number of base pairs is possible when the first bases pair with the last bases of the RNA sequence. In our implementation, we took some threshold values:

Considering the two position of bases 'p' and 'q' and length of RNA sequence as 'l', we choose base pairs (bp) with the following conditions:

$$bp = \left\{ \begin{array}{ll} (l-3) < (p+q) < (l+3), & l < 10 \\ (l-6) < (p+q) < (l+6), & l < 400 \\ (l-10) < (p+q) < (l+10), & l > 400 \end{array} \right\} \tag{4.3}$$

We observed that, the number of MIS generated is independent of the sequence length. However, a large number of MIS might be generated for a sequence rich in AT or GC bases, possibly because they lead to a large number of base pairs.

4.3 A review on RNA secondary structure prediction using deep learning methods

Apart from the classical methods, machine learning and deep learning based methods have recently gained wide attention to predict RNA secondary structures. In this section, we have reviewed these available methods and software implementations for research purposes.

4.3.1 Computational methods that predict pseudoknot-free structures

4.3.1.1 Learning based RNA folding methods

Bases in the stem region of RNA secondary structure are nearly palindromic. Therefore, Context-Free Grammars (CFG) have been used along with machine learning approaches in predicting RNA secondary structures. As the name suggests, in stochastic context-free grammar (SCFG) based algorithms, probabilities are appended to the context-free grammar production rules used for secondary structure prediction. Production probabilities are learned from parse trees or directly from the RNA sequences. Training algorithms such as the inside-outside method have proven to provide consistent SCFGs [103][96]. Apart from that, conditional log-linear models, along with Maximum expected accuracy, has also been devised with SCFG using machine learning approaches to predict secondary structure. CONTRAfold [37] and MaxExpect [122] are examples of such approaches.

4.3.1.2 Deep-learning based RNA folding methods

Recently deep neural networks have also been used to predict pseudoknot-free structures. NNFold [225], CDPFold [247], MxFold2 [182] or using the nearest

neighbour model with deep learning [236] are examples of such approaches. In CDPFold the CNN has been trained over large datasets containing known RNA secondary structures, and implicit features are retrieved. The output layer of CNN predicts the probability of each base with respect to three dot-bracket notation symbols '(', ')' and '.'. To resolve a few discrepancies like: the unequal number of left and right brackets, mis-pair of suggested matched brackets with the corresponding bases, maximum probability sum algorithm (MPSA) [249] is used. This MPSA algorithm is an extended version of Nussinovs dynamic Programming algorithm [139].

4.3.2 Computational Methods that predict pseudoknotted structures

4.3.2.1 Machine learning methods to predict pseudoknotted structures

Core machine learning algorithms like SVM are also used to predict secondary structure [77][88], based on the principle that the different RNAs have different properties and different machine learning tools can identify different properties of secondary structure. Other machine learning techniques use gradient-based methods that use the derivatives of the thermodynamic model against pairing probabilities [22]. A decision tree (DT) based method along with a training algorithm has been proposed. Maintaining a window size with decision trees are used to realize the pseudoknots [118], and machines are also developed with multi-layer perceptron to predict the pseudoknots [121].

4.3.2.2 Deep learning methods that can predict RNA secondary structure with pseudo-knots

The discipline of machine learning algorithms has been revolutionized by deep neural networks. It has become a standard tool for computer vision, natural

language processing or speech recognition problems. Deep neural networks consist of many layers that map the data from input space to output space, thus learning higher level features of the training data. Recently, deep neural networks are also applied in predicting RNA secondary structures. The deep learning models vary among themselves with respect to architecture, model of input, and output formats (Table 4.4).

(i) **RNN_App** : The Recurrent Neural Networks (RNN) [235] models came into the light to predict RNA secondary structure. Input to this network is a one-hot encoded vector. The input with its cartesian product is passed to Bi-directional Long Short Term Memory (BLSTM). Several hidden layers of CNN are then applied, which are passed to the output layer. The output is an LxL (L is the length of RNA sequence) matrix with pairing probabilities within the interval [0,1]. To check for overlapping base pairs, the pair let's say (i, j) with the highest probability is first chosen and is put in a set S representing the final secondary structure. Then the probabilities of all bases that are associated with either i or j are set to 0. The pair with the next highest probability is then chosen until all the bases in the RNA sequence are traversed.

(ii) **SPOT-RNA**: The architecture of this SPOT-RNA model [197] uses ResNets, BLSTM, FCL, and transfer learning techniques. Input to the architecture is a one-hot encoding which is then extended by self-Cartesian product. Two datasets considered in this model are namely bpRNA and PDB. The five best models were chosen using the training and validation data from many 2D learning models. The PDB dataset contains a small number of RNA sequences; hence it is insufficient to train the network. So, transfer learning is applied, where all the models are first trained over larger dataset i.e. bpRNA dataset and then the learning parameters like weights and biases are stored in the model. The model is then trained over a smaller dataset i.e. PDB dataset. The advantages of applying transfer learning are: it takes less amount of

Table 4.4: Deep learning architectures that predict pseudoknotted RNA Secondary Structure

Sl No.	Alias	Author	Year	Model components	Dataset used	Resource	Reference
1	RNN_App	D. Willmott	2018	BLSTM, CNN	RNA STRAND	http://github.com/dwillmott/ss-inf	[235]
2	SPOT-RNA	Singh et al.	2019	ResNet, BLSTM, FCL, TL	bpRNA, PDB	https://github.com/jaswinder singh2/SPOT-RNA/	[197]
3	DMfold	Wang et al.	2019	BLSTM, FCL, IBPMP, CSCP	ArchiveII	https://github.com/linyuwangPHD/ RNA-Secondary-Structure-Database	[232]
4	Adaptive DRNN	Lu et al.	2019	Adaptive deep RNN, LSTM, EBF	RNA STRAND	http://eie.usts.edu.cn/prj/ AdaptiveLSTMRNA/index.html	[119]
5	DpacoRNA	Quan et al.	2020	LSTM, PACO	RNA STRAND	–	[157]
6	2dRNA	Mao et al.	2020	BLSTM, FCL, CNN, U-net	ArchiveII	https://biophy.hust.edu.cn/new/2dRNA	[126]
7	E2Efold	Chen et al.	2020	Transformer Encoder, CNN, Unrolled algorithms	ArchiveII, RNAStralign	https://github.com/ml4bio/e2efold	[30]
8	Ufold	Cao et al.	2020	Base pairing map, CNN, contact score map	RNAStralign, ArchiveII, bpRNA-1m, bpRNA-new	https://github.com/uci-cbcl/Ufold	[23]
9	ATTfold	Wang et al.	2020	Transformer Encoder, CNN	RNAStralign	–	[234]
10	SPOT-RNA2	Singh et al.	2021	PSSM, DCA, DCN, ResNet, FCL, TL	bpRNA, PDB	https://github.com/ aswindersingh2/SPOT-RNA2	[198]
11	GoogLeNet and TCN	Shen et al.	2021	GoogLeNet, TCN, CNN, DP	ArchiveII	–	[196]
12	VLDB GRU	Lu et al.	2021	GRU, FCL	RNA STRAND	–	[120]
13	2dRNA-LD	Mao et al.	2022	Transfer Learning, CGDBP, FGDPP, LSTM, U-net	bpRNA, PDB	https://biophy.hust.edu.cn/new/2dRNA	[127]
14	RSSM	Yan et al.	2022	Transfer learning, covariance features, transformer	Rfam, bpRNA	–	[245]

time on training the second dataset, the performance of the network model increases, also the second dataset need not be very large. The dataset over which the network was trained consists of a sequence length less than or equal to 500 bases. The reason being hardware limitations and high-resolution RNA of length more than 500 is not abundant.

(iii) **DMfold**: This model [232] uses BLSTM, FCL and a probability method named as Improved Base Pair Maximization Principle (IBPMP). DMFold uses 3975 RNA sequences with known primary structure from public database of Mathews Lab. Data cleaning for redundant or similar sequences is done by CD_HIT. There are two parts of DMFold: Prediction Unit (PU) and Correction Unit (CU). The Prediction Unit (PU) part has two segments, encoder and decoder. Encoder is used to encode each base of RNA sequence. As the secondary structure of RNA is assumed to be context-dependent, three layers of Bidirectional Long Short-Term Memory (BLSTM) is used as encoder, which uses its memory capacity to remember contextual information of the sequence. The forward LSTM processes RNA sequences from left to right and backward LSTM in reverse order. The Correction Unit (CU) part is for compensating any errors of PU part and thus predicts RNA structure using Improved Base Pair Maximization Principle (IBPMP). IBMP is a modified measure used to select candidate stems in multiple steps and assign different priorities to them to determine secondary structure of RNA. CU part first obtains all possible pseudoknot-free structures and then determines the optimal pseudoknotted structure. The method works well for shorter sequences, but mediocre for longer sequences. Dataset for training is not very large. RNA sequences vary in length, so a threshold of 300 bases is kept and the sequences are truncated, also if it is less than 300, a series of N is padded to the sequence, which may lead to loss of information.

(iv) **Adaptive DRNN**: Machine learning methods employed to determine secondary structure of RNA uses fixed size feature set, which leads to truncation

of sequences into parts and then to use neural network model on that part which may lead to loss of information. So, an adaptive sequence length based on deep learning model is used in ADNN [119]. The architecture of ADNN consists of three modules: Adaptive Module, LSTM module and Energy-based filter. Adaptive model is used for adaptability of the model of variable length sequence. If the maximum length, the model could handle is L and the sequences size are less than L, then the feature set of sequences is extended with arbitrary feature values up to L. A mask vector is used of size L, where the elements of mask vector are 1 if the feature is of original sequence, 0 for extended sequence. If the mask value is 0 for a base the gradient value will be 0 and hence for extended sequence weights will not be updated. Along with mask vector a dynamic weight vector is also used, the weight for any corresponding base would be equal to the number of unpaired bases if the base is paired in the original sequence, otherwise 1. For the LSTM module, the input features are mapped into higher dimensional feature vectors and inputs them to both forward and backward LSTM, the output is then passed to fully connected layer, and then an output layer with softmax activation function is used to determine the classification probabilities or the pairing results. LSTM may result in conflicting base pairs, so an energy-based filter is used to address this issue, the conflicting base pairs are extracted and a matrix is generated which is initialized with 0s and 1s randomly. Then for each combination thermodynamic energy is calculated, the one with minimum energy values is retained as the final structures. And then the output is generated as pairing result.

(v) **DpacoRNA**: Parallel ant colony optimization (PACO) algorithm augmented to deep learning method is also used for searching motifs of secondary structure from primary sequence of RNA [157]. Deep learning model is specifically used to learn structural constraints of secondary structure. PACO is used to increase ability of global optimal search. It has the ability to run

in parallel with different objective functions, where they share a matrix that can be used to share experiences learned from different ant colonies. Using different ant colonies help to incorporate multiple objective functions. A deep learning model is used to learn the structural constraints and then to use them to filter out discrepancies of PACO.

(vi) **2dRNA**: The network model [126] is a two-stage model, the first stage is for determining if the base is paired or unpaired. In the next stage the errors of mispairing are compensated using U-net segmentation model. The first stage is termed as coarse-grained dot-bracket prediction (CGDBP), which takes input as one-hot encoding of RNA sequence and passes it to two layers of bidirectional LSTM, which is then passed to 3 layers of fully connected layer that gives the output in one-hot encoded vector of output notations. The next stage is fine-grained dot-plot prediction (FGDPP) which takes the output of CGDBP and passes it to LSTM and a fully convolutional network U-net to obtain the correct base pairings.

(vii) **E2Efold**: This network [29] is also a two-stage model. The first stage uses 3 layers of transformer encoder over the positional encoding of RNA sequence. It concatenates the results and passes on to 2 layers of CNN that outputs a score matrix. This stage encodes any complex information or dependency of sequence. In the next stage, the constraints of RNA are gradually enforced using unrolled algorithm. The network model is an end-to-end model, which enforces the constraints during the training procedure.

(viii) **Ufold**: This method is based on an encoder-decoder transformer architecture [23]. Input to this architecture is an element wise cross multiplication of a one-hot encoded vector of RNA sequence of size Lx4 to its transpose. Each row of one-hot encoded matrix is cross multiplied with its transpose, leading to 16 such matrices. Thus, the model considers all long-range relationships of base pairs. In this encoder decoder architecture, the encoder derives the semantic representation of input and decoder fills the contextual information.

The output of the architecture is a scoring matrix that consists of base pair probabilities. The issue of overlapping base pairs is resolved by linear programming. The final output is the predicted secondary structure in dot bracket notation.

(ix) **ATTfold**: This network model [234] employs the transformer encoder to retrieve the global information of sequences using the attention mechanism. To avoid the problem of vanishing gradient, Residual Networks are employed. The output of the encoder is a symmetric matrix. It then employs CNN as a decoder which takes the symmetric matrix of the encoder as input and finally produces a base pair probability matrix. In the next step, the hard constraints of the RNA secondary structure are enforced into the network to obtain the final prediction.

(x) **SPOT-RNA2**: This network model [198] takes one hot encoded vector of RNA and features of homologous sequences from NCBI database and from SPOT-RNA as input. The features extracted from homologous sequences are termed as Position specific scoring matrix (PSSM), and mutational direct coupling analysis (DCA). These features are then passed to the deep neural network architecture, which is comprised of Dilated convolutional networks (DCN). This network is reported to be faster and better than the LSTMs to cover long range dependencies. Transfer learning technique is used to train the model in a larger data set and then to smaller data set to produce better results.

(xi) **GoogLeNet and TCN**: The network [196] is a two-step model. In the first step, GoogLeNet and Temporal convolutional network (TCN) is used to determine the base probability. Both the networks use convolutional neural network to build the architecture. In GoogLeNet, Inception network model is used that reduced the dimension of the network and number of parameters. To avoid the vanishing gradient problem softmax layers are incorporated. While in TCN, casual, dilated convolution layer and residual blocks are

employed. The method gave more importance to the bases of the stem that appear in the centre of the stem as compared to those appearing at the edges. In the second step, correction of base pairs is taken care. By using dynamic programming and following the rules of secondary structure, the correct base pairings of the secondary structure is obtained.

(xii) **VLDB GRU**: The features of the model [120] are taken as the base pairing probabilities using partition function. The model employs recursive bi-directional gated recurrent unit (GRU) to capture long range dependencies. Instead of truncating the sequences, it appends 0s as a flag vector and these values are taken care by the loss function and is not considered in the training phase. A weight vector is introduced to maintain the imbalance of paired and unpaired bases in the RNA sequences. The model maps the feature vector to the input of the bidirectional GRU. The output of GRU is followed by 2 FCL and an output layer. Drop outs are added to overcome overfitting.

(xiii) **2dRNA-LD**: The network model [127] is a two-step process similar to 2dRNA. The model is an ensemble of the best five models chosen on the basis of hyperparameters trained over the training set. Then the training dataset is split into different sets having proportionate length. Transfer learning is then applied using the model of the first step to all the sets of the training data, hence they termed the model as length dependent (LD).

(xiv) **RSSM**: The network model [245] uses evolutionary information to predict single sequence structure and transfer learning from pre-trained models using multiple sequence alignment and evolutionary information. For longer sequences, a sliding window method is used that is based on local structure prediction method. MSA Transformer is used that detects the base pair probability from homologous sequences.

(xv) **Other deep learning techniques**: Approaches have also been used to predict secondary structure of RNA comprising LSTM, ResNet [233]. Au-

Table 4.5: Confusion Matrix

	BPPredicted: **YES**	BPPredicted: **NO**
BPActual: **YES**	True Positive (TP)	False Negative (FN)
BPActual: **NO**	False Positive (FP)	True Negative (TN)

toencoders and neural networks based on graphs have also been used to predict secondary structures of RNA [24][244]. Deep learning techniques along with thermodynamics and architectures like transformers have also been used to predict secondary structure of RNA [182][29].

4.3.3 Comparative Results and Discussion

4.3.3.1 RNA secondary structure information

For a comparative study among pseudoknotted RNA secondary structure prediction methods, we considered 25 RNA sequences whose secondary structures are available in the public domain in bpRNA [33] and URS database [11]. The length of pseudoknotted structures ranges from 28 to 406 bases. In this study, we have considered only smaller RNA sequences (of size less than 500 bps) because available software tools can efficiently process sequences of this size (Appendix A.3.1).

4.3.3.2 Performance measurement

Confusion matrix and associated matrices, are commonly used for evaluating performance of the prediction methods [81][107][111][253]. Confusion matrix is an MxM matrix where M is the number of classes. In the secondary structure prediction problem, any base in the RNA sequence can be grouped into two classes, (i) paired and (ii) unpaired and accordingly, confusion matrix can be presented as in Table 4.5 where BP denotes base pair.

However, when the pairing position of a base differs between actual and predicted structure, there is no consensus among existing literature

Table 4.6: Anomalous Representation

Positions	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**	**12**	**13**	**14**	**15**	**16**	**17**	**18**
RNA sequence	G	A	C	G	G	A	A	U	U	C	U	C	C	A	A	G	U	C
Actual Structure	(	(	.	(	(	.	.	[	)	)	(	.	.	.	)	]	)	)
Predicted Structure	(	.	(	(	.	.	.	)	[	(	(	.	.	]	)	)	)	)
Actual Structure (BPSEQ)	18	17	0	10	9	0	0	16	5	4	15	0	0	0	11	8	2	1
Predicted Structure (BPSEQ)	18	0	17	8	0	0	0	4	14	16	15	0	0	9	11	10	3	1
Confusion Matrix Value	TP	FN	FP	**FP**	FN	TN	TN	**FP**	**FP**	**FP**	TP	TN	TN	FP	TP	**FP**	**FP**	TP

regarding values to be filled in. In this study if a base pair is predicted wrongly, we consider the prediction as FP. This anomaly is explained with the hypothetical example Table 4.6. For example, let us have a hypothetical sequence of length 18 bp, and the actual and predicted secondary structure in the dot-bracket notation is given in Table 4.6. The anomalies can be seen in position 8,9,10 etc. A clearer picture can be visualized using the BPSEQ format of both the actual and predicted structure along with the remark as given in Table 4.5. We can infer from Table 4.5 that; TP is marked where index of a base and its corresponding base pair matches as in (1,18) in both actual and predicted structure. TN is marked if for a base, corresponding pairing value is 0 (i.e., unpaired) as in (6,0) in both actual and predicted structure. FN is marked if in actual structure base pair is predicted as in (2,17) but not in predicted structure (2,0). FP is marked if the actual structure has an unpaired base as in (3,0) but the predicted structure marked it as paired as in (3,17). Apart from these, base pairs in actual structure as in (4,10) and in predicted structure as in (4,8) do not match. In this study, these wrongly predicted base pairs are considered as FP and marked bold in Table 4.6.

4.3.3.3 Performance metrices

To compare the actual RNA structures with the one obtained from predicted model, we have used performance metrices namely Accuracy, Precision and F1 Score as defined below: Accuracy measures fraction of predictions, model

could correctly predict a class:

$$Accuracy = ((TP + TN))/(TP + TN + FP + FN) \tag{4.4}$$

Precision measures the correctly predicted base pairs among all base pairs that exist in predicted structure.

$$Precision = TP/(TP + FP) \tag{4.5}$$

Recall measures the correctly predicted base pairs among all base pairs that exist in actual structure.

$$Recall = TP/(TP + FN) \tag{4.6}$$

F1 Score is the harmonic mean of Precision and Recall values. F1 Score value can be between 0.0 and 1.0. A good prediction model would be one whose F1 score is high (close to 1.0). F1 score is high when both Precision and Recall values are high.

$$F1\ \ Score = 2 * (Precision * Recall)/(Precision + Recall) \tag{4.7}$$

We calculated Accuracy, Precision and F1 Score for all 25 RNA sequences. Since we classified the wrongly predicted base pairs into FP, Precision will be a critical measure to compare among the models.

4.3.3.4 Results

Based on the availability of implementation and scope for predicting new RNA structures, we have compared two methods namely SPOT-RNA [197] and UFold [23] for 25 pseudoknotted structures. We have used CT formats of both the methods to compare between the structures and the gene wise metrices are present in Appendix A.3.2.

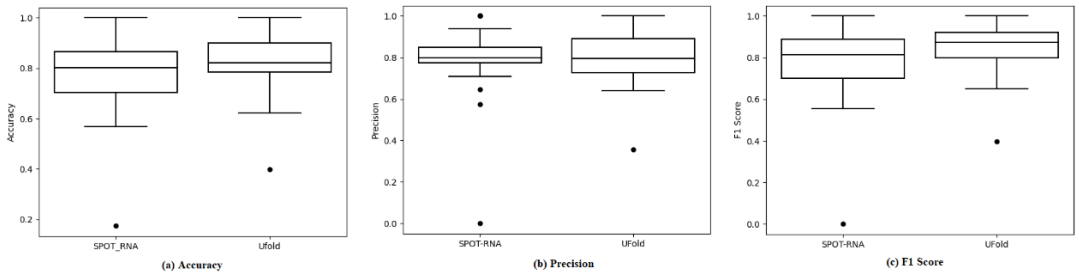

Figure 4-3: Box-plot of Accuracy, Precision and F1 Score for pseudoknotted structures

Box plot representations for methods that predict pseudoknotted structures are presented in Figure 4-3. Accuracy is depicted in Figure 4-3(a). IQR is small in case of Ufold (0.12) as compared to SPOT-RNA (0.16) and Q1 is also higher (0.78) in Ufold as compared to SPOT-RNA (0.70). In terms of Accuracy, Ufold is a better measure. But, the number of instances in paired and unpaired class is not equal, as 2/3rd of the secondary structure of RNA constitutes of paired region and 1/3rd of unpaired regions. So, we further observed the performance of these methods using Precision and F1 Score.

The precision values are presented in Figure 4-3(b). IQR is small in case of SPOT-RNA, also the values are skewed towards higher precision score where Q1 is at 0.77. SPOT-RNA has a smaller number of wrongly predicted base pairs with respect to all predicted base pairs, whereas in case of Ufold, the IQR is comparatively larger, also the Q1 value is less (0.72).

The F1 scores of the methods are displayed in Figure 4-3(c). Smaller IQR is in Ufold as compared to SPOT-RNA, also the Q1 values are higher in case of Ufold (0.92) than SPOT-RNA (0.64). Hence in terms of F1 Score, Ufold is a better prediction method.

Appendix A.3.3 presents the Count of RNAs, mean, standard deviation, minimum value, 25% or Q1, 50% or Q2, 75% or Q3, maximum value, Interquartile range (IQR) of Accuracy, Precision, Recall, Specificity and F1 Score of the pseudoknotted structures analysed.

4.3.4 Conclusion

Machine or deep learning techniques have been extensively explored in recent years for predicting RNA structure. Models based on these techniques require a large amount of data to train for better prediction result. At present the datasets available in public domain contain huge number of RNA structures, but after pre-processing and removing the sequences having higher similarity, very few unique sequences remain to train the deep neural network. Algorithms that depend on homologous sequences for prediction accuracy might be sensitive to alignment errors. Further, issue like the class imbalance between paired and unpaired bases which may lead to locally optimal results. When RNA sequence length differs in training dataset and the machine learning algorithm accept only fixed length data, truncating or adding arbitrary data to the sequences may result in loss of information or reduce accuracy. Machines are yet to be trained that could identify complex pseudoknots. Non-canonical base pairs or triplets may also exist in the RNA secondary structure. Techniques to introduce these pairs without much increasing the complexity of the program needs to be proposed. In addition to these model related issues, future scope lies in improving software implementations. Most of the software implementations accept RNA sequences of length only up to 2000 bases which may not be sufficient. Often symbols like (), [], {}, aA, bB, cC etc., are used while representing RNA structures with pseudoknots. Uniform standard annotations are to be adopted while representing these motifs.