Software Design Document

This Software Design document gives out explanation of various methods seen in dfferent .py files.

Table of Contents

Methods

Purge

Overview:

The purge method will take in the input and remove data that satisfy the following conditions:

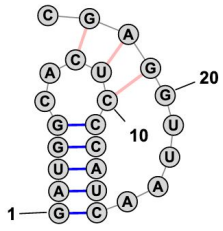
1. The sequence has less than 300 nucleotide bases.

Input[1]:

The input file has the following format rules:

1. Sequences are case sensitive.
2. Capital letters are used in nominal situations.
3. Lower case letters indicate a base that cannot form base pairs (i.e. is constrained to be single-stranded).
4. Sequences can contain U or T interchangeably.
5. White spaces (space, tab, line breakers) are allowed and ignored.
6. “X” and “N” are used to represent an unknown base or a base that cannot interact with other bases.
7. Three or more consecutive “X” and “N” indicate an unstructured loop, which can represent a section of unknown identity or a section that has been purposely left out of the prediction.
8. The “.” symbol represent an unpaired nucleotide
9. An open-parenthesis "(" represents the 5'-nucleotide in a pair, and the matching closing parenthesis ")" represents the 3'-nucleotide in the pair.
10. Caret symbols are used to represent pseudo-knots (figure 1)

```
>A pseudo-knot structure
GAUGGCACUCCCAUCAAUUGGAGC
((((((..<<<)))))).....>>>.
```



Output

The output is a list with N arrays. Each array contains two elements as the following:

The first element is the RNA sequence.

The second element is the associated RNA secondary structure in the dot-parentheses representation.

[1] https://rna.urmc.rochester.edu/Text/File_Formats.html

Sequence Tokenizer

Overview:

The Keras library provided tokenizer is used to perform the following steps:

1. Tokenize the parsed sequences with a length of k.
2. Train and learn to fit the sequence to unique integer representations.
3. Convert the sequences from letter representation (‘A’, ‘C’, ‘G’, ‘T’) to integer representation.
4. Save the integer representation to original sequence mapping.

Input

The input is a list that contains arrays of the RNA sequences in the letter representation format.

Output

1. A list that contains the RNA sequences in the integer representation format.
2. A set that contains unique letter to integer representation mapping.

Feature and Label Extractor

Overview:

The RNN needs a defined input array dimension. The feature and label extractor will perform the following operations:

1. The RNA primary sequence will be added to a list as the feature
2. The RNA secondary structure will be added to a list as the label
3. The feature list will be transformed to be an N by M matrix
4. The label will be one-hot coded and transformed into a matrix with dimension M by L, where L is the length of the vocabulary (unique “words” from the tokenizer)

Input

A list that has the information for both the RNA primary sequence and the RNA secondary structure

Output

1. A feature matrix with dimension N by M
2. A label matrix with dimension M by L

RNN Builder

Overview:

The RNN Builder will utilize the Keras library to build the RNN with the following configurations[2] , which can also be found at the Keras website:

1. Embedding layer:
    a. Turns positive integers (indexes) into dense vectors of fixed size.
2. Recurrent layer:
    a. a single layer of LSTM cells with dropout to prevent overfitting.
3. Fully connected layer:
    a. This layer adds additional representational capacity to the network.
4. Dropout regularization:
    a. to prevent overfitting for the training data.
5. Output layer:
    a. This produces a probability for every word in the vocab using specified activation.
6. Compile:
    a. defines the optimizer, loss function, and metrics.
7. Callback and model check point:

Input

1. The column dimension of the input feature matrix.
2. The column dimension of the input label matrix (the length of the vocabulary)

Output

The training history.

[2] https://keras.io/getting-started/sequential-model-guide/