5-2018

End-to-end learning framework for circular RNA classification from other long non-coding RNAs using multi-modal deep learning.

Mohamed Chaabane
University of Louisville

Recommended Citation
Chaabane, Mohamed, "End-to-end learning framework for circular RNA classification from other long non-coding RNAs using multi-modal deep learning." (2018). *Electronic Theses and Dissertations.* Paper 2954.
https://doi.org/10.18297/etd/2954

END-TO-END LEARNING FRAMEWORK FOR CIRCULAR RNA CLASSIFICATION FROM OTHER LONG NON-CODING RNA USING MULTIMODAL DEEP LEARNING

By

Mohamed Chaabane
B.E., Computer Science , Tunisia Polytechnic School, 2016

A Thesis
Submitted to the Faculty of the
J.B. Speed School of Engineering
in Partial Fulfillment of the Requirements
for the Degree of

Master of Science in Computer Science

Department of Computer Engineering and Computer Science
University of Louisville
Louisville, Kentucky

May 2018

END-TO-END LEARNING FRAMEWORK FOR CIRCULAR RNA CLASSIFICATION FROM OTHER LONG NON-CODING RNA USING MULTIMODAL DEEP LEARNING

By

Mohamed Chaabane
B.E., Computer Science , Tunisia Polytechnic School, 2016

A Thesis Approved On

4/23/2018
Date

By the following Thesis Committee:

Juw Won Park, Ph.D., Thesis Director

Eric C. Rouchka, Ph.D.

Donghoon Chung, Ph.D.

ACKNOWLEDGEMENTS

I would like to thank my advisor Dr. Park for his continuous support, friendly assistance and enthusiastic encouragements during both planning and fulfillment of this work.

I would also like to thank Dr. Eric C. Rouchka and Dr. Julia Chariker for insightful discussions and comments.

I address, likewise, my thanks to my colleagues in the Computer Engineering and Computer Science Department for their support and friendship. Finally, I want to convey my most heartfelt thanks to my family for their continuous support and unconditional love.

ABSTRACT

END-TO-END LEARNING FRAMEWORK FOR CIRCULAR RNA CLASSIFICATION FROM OTHER LONG NON-CODING RNA USING MULTIMODAL DEEP LEARNING

Mohamed Chaabane

April 11, 2018

Over the past two decades, a circular form of RNA (circular RNA) produced from splicing mechanism has become the focus of scientific studies due to its major role as a microRNA (miR) activity modulator and its association with various diseases including cancer. Therefore, the detection of circular RNAs is a vital operation for continued comprehension of their biogenesis and purpose. Prediction of circular RNA can be achieved by first distinguishing non-coding RNAs from protein coding gene transcripts, separating short and long non-coding RNAs (lncRNAs), and finally predicting circular RNAs from other lncRNAs. However, available tools to distinguish circular RNAs from other lncRNAs have only reached 80% accuracy due to the difficulty of classifying circular RNAs from other lncRNAs. Therefore, the availability of a faster, more accurate machine learning method for the identification of circular RNAs, which will take into account the specific features of circular RNA, is essential in the development of systematic annotation. Here we present an End-to-End multimodal deep learning framework, our tool, to classify circular RNA from other lncRNA. It fuses a RCM descriptor, an ACNN-BLSTM sequence descriptor, and a conservation descriptor into high level abstraction descriptors, where the shared representations across different modalities are integrated. The experiments show that our tool is not only faster compared to existing tools but also eclipses other tools by an over 12% increase in accuracy. Another interesting result found from analysis of a ACNN-BLSTM sequence descriptor is that circular RNA sequences share the characteristics of the coding sequences.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

CHAPTER 1

INTRODUCTION AND BACKGROUND

Non-coding RNA (ncRNA) [1] is functional RNA that is transcribed from DNA, but is incapable of being translated into protein. ncRNAs can be categorized into two groups based on length; short non-coding RNAs, which are shorter than 200 nucleotides, and long non-coding RNA (lncRNA), which are longer than 200 nucleotides. lncRNAs have been proven to have a critical duty in several cellular functions; e.g., protein synthesis in a multitude of distinct processes and gene regulation [2] and the development and pathophysiology of disease [3].

The evolution of next-generation-sequencing has created an excess of sequencing data, which has led to an increased discovery of lncRNAs. Experimentally identifying and annotating each of these new sequences is infeasible due to the large amounts of time and funds it would take to do so. Therefore, computational methods are required to analyze and find potential new lncRNAs candidates in the genome. There are presently a multitude of superb solutions to differentiate lncRNA from protein coding RNA with precision for constructed transcripts from modern sequencing. In the past, solutions have only concentrated on distinguishing lncRNAs from protein coding RNAs, however there are several kinds of lncRNAs in the human genome. lncRNA can be approximately segregated into antisense, processed transcript, sense intronic, and sense overlapping in GENCODE [4].

A sub category of lncNRAs, circular RNA, has become the focal point of scientific studies over the last two decades in various species due to its correlation with a myriad of diseases including cancer [5–7] and its vital function as a microRNA(miRNA) activity modulator [8,9]. Circular RNA is produced by ligating a downstream donor site (5' splice site) of the flanking downstream intron with an upstream acceptor site (3' splice site) of a second upstream intron; this process is a unique type of alternative splicing, referred to as back-splicing. This is contrasted by canonical alternative splicing which joins an upstream donor site (5' splice site) with a downstream acceptor site (3' splice site) within a single intron, and produces a linear configuration of RNA (Fig. 1.1).

Recently, there has been a growing number of distinguished circular RNAs; it has been

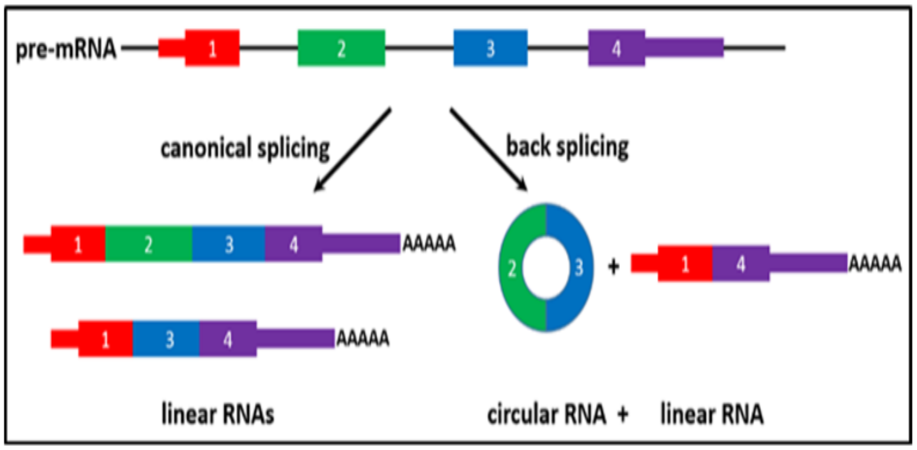

Figure 1.1: Illustration of linear RNA and circular RNA formation

estimated that circular RNAs are assembled from over 10% of genes [10]. Circular RNAs have a greater stability than linear RNAs due to their structure which excludes 5' and 3' ends. Circular RNAs are also immune from exonuclease mediated degradation.

As of now, the majority of the circular RNAs have not been established to aid in protein coding. Therefore, it can be assumed that this category of functional RNAs can synonymously be regarded as noncoding RNAs. It is difficult to discern between cirRNAs from other lncRNAs even though the creation of cirRNAs is significantly different from lncRNAs. Each category of lncRNA has distinct traits and roles, so there is a large benefit to pinpointing various categories of lncRNAs. There are already computational solutions to classify small ncRNAs into various categories but there is a lack of the same solutions for lncRNAs.

The recognition of circular RNAs is a vital operation for continued comprehension of their biogenesis and purpose. A substantial amount of circular RNAs have been annotated in the transcriptome with the advent of modern experimental technology. Unfortunately, it remains an extensive problem to recognize circular RNAs from traditionally labeled lncRNAs due to the strenuousness of experimental data analysis and the low expression that almost all lncRNAs have [11].

As of now only two tools are accessible for computational categorization of circular RNA from other lncRNA. The first is PredcircRNA [12] which is a computational approach based on a multiple kernel learning framework trained with a variety of features; i.e., graph features, component composition features, conservation score features, features of ALU and tandem repeats, the ORF, and SNPs from transcripts. The second is H-ELM and it extracts identical features and categorizes circular RNAs from other lncRNAs by utilizing a hierarchical extreme learning machine (H-ELM) algorithm with feature selection [13]. For the dataset proposed [12], PredcircRNA reached 0.778 accuracy with 0.554 Matthews correlation coefficient (MCC) and H-ELM reached 0.789 accuracy with 0.561 MCC. Neither of these methods are perfect. Both of them have non-negligible drawbacks that could prevent them from being reliably adapted by the bioinformatics community. Neither succeeds at using features that describe the unique structure of circular RNA and both methods use sequence information with trinucleotide frequencies but fail to employ the co-occurrence relationship of trinucleotide.

To address the limitations of existing approaches, we propose circDeep, an end-to-end machine learning framework for robust circular RNA prediction. In this work, we introduced an innovative feature descriptor that we called Reverse Complement Matching (RCM) descriptor which aims to extract the potentiality of the flanking sequences to the query sequence to make the circularization process. We propose also another feature descriptor that we called ACNN-BLSTM sequence descriptor which combines the asymmetric convolution neural network (ACNN) with the Bidirectional Long Short Term Memory network (BLSTM) which is able to extract from each sequence local patters as well as the long-range dependencies. These two novel descriptors are fused with a conservation descriptor which is composed of features that contain information about conservation of a specific sequence among species as well as conserved motifs. Finally, in order to fuse different heterogeneous descriptors, we propose a deep architecture to construct the nonlinear representation from different aspects of information sources. To the best of our knowledge, circDeep is the first method that uses deep model for circular RNA prediction.

CHAPTER 2

RELATED WORK

2.1 Circular RNA classification from other long non-coding RNA

As of now only two tools are accessible for computational categorization of circular RNA: PredcircRNA [12] and another paper that uses a hierarchical extreme learning machine [14]. PredcircRNA used various machine learning models to learn a linear classifier based on scalar features. The models rely on external software to extract these features from the RNA transcripts.

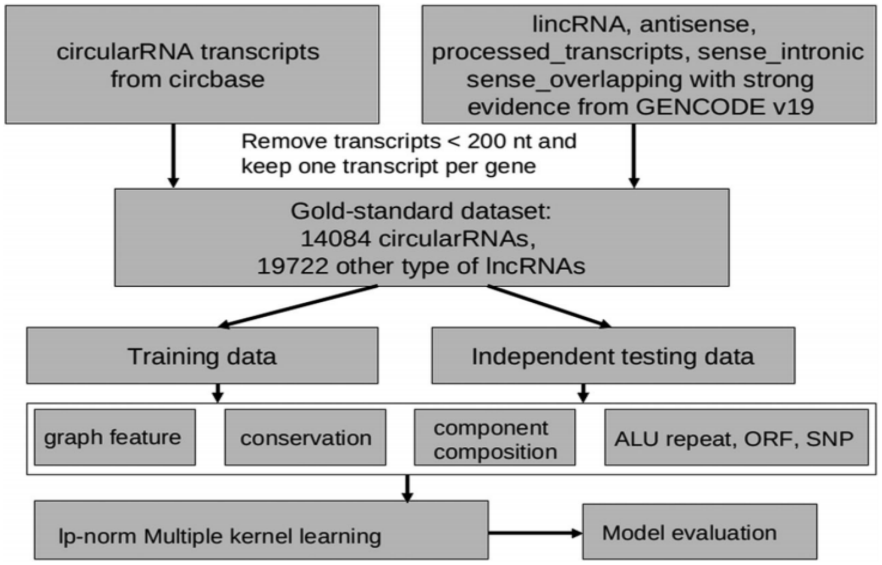
circularRNA transcripts from circbase
lincRNA, antisense, processed_transcripts, sense_intronic sense_overlapping with strong evidence from GENCODE v19
Remove transcripts < 200 nt and keep one transcript per gene
Gold-standard dataset: 14084 circularRNAs, 19722 other type of lncRNAs
Training data
Independent testing data
graph feature
conservation
component composition
ALU repeat, ORF, SNP
lp-norm Multiple kernel learning
Model evaluation

Figure 2.1: PredcircRNA's pipeline

A large portion of these features were extracted using GraphProt which uses graph kernels to obtain over 30,000 graph features. PredcircRNA uses random forests to discover which of these are the most important for circular RNA classification, and took the top 101 features to use as inputs to the machine learning model. Other scores derived from the RNA sequences are used to classify the RNA such as conservation scores which make up another 12 of the features. Trinucleotide scores that measure the frequency of various trinucleotides in the RNA transcript are also extracted in addition to other nucleotide composition features including GC, AG, GTAG and others. ALU and tandem repeats, ORF (open reading frames), and SNP (single nucleotide polymorphism) were used as features for the model as well, and are known to be discriminating for lncRNA. Miscellaneous features such as sequence length are also included. Although it does not discriminate circular RNA well on its own, any additional features contribute information and increase the model's ability to discriminate circular RNA successfully.

For validation of the model, the authors used a subset of the circBase database as a positive dataset of circular RNA transcripts, consisting of the 14,804 transcripts longer than 200nt. They used GENCODE as the negative dataset to obtain 19,722 negative samples, after removing overlaps with circBase and selecting only higher confidence level RNA transcripts.

PredcircRNA compares support vector machines (SVM), random forests, and multiple kernel learning (MKL) at the task of classifying circular RNA from the extracted features. All of them obtain similar results, reaching 76-78% accuracy on the withheld testing data. This indicates that the classification accuracy was limited by the extracted features, not by the ability of the machine learning model used to fit the data. Multiple kernel learning achieved the highest score on the circular RNA classification task, but was only very slightly more accurate than the support vector machine.

Of the features collected, they sort them by importance using random forests and analyze the top 50. Conservation and graph features make up most of the top features. Length comes in at 8th place, marking it as fairly important. A few trinucleotide frequency features also make it into the top 50, such as CAA and TTG.

They continue their experiment by performing multiclass classification on circular RNA, lincRNA (long intergenic noncoding RNA), antisense, and processed transcript. They used the one-vs-another technique on a balanced dataset consisting of 2700 samples for each class. They achieved an overall accuracy of 0.604, with circular RNA being the most easily confused for other classes and lincRNA being the most likely to be missclassified as circular RNA.

They studied the model's performance when restricted to only use a subset of the extracted features. Conservation features alone allow for 0.703 accuracy, indicating they best distinguish circular RNA from other lncRNA. Besides accuracy, they measured sensitivity, specificity, precision, and MCC . Component composition leads to the best sensitivity. Conservation features give the best specificity as well as accuracy. ALU and tandem repeat features obtain the best precision. With these statistical advantages of each set of features, the model is able to perform best by combining all of them.

Lei Chen et al [13] continues the work done in PredcircRNA by applying H-ELM (Hierarchical Extreme Learning Machine) with feature selection to to the same features used in PredcircRNA. Where PredcircRNA used random forests to sort the features by importance, they use minimum Redundancy Maximum Relevance (mRMR). mRMR creates a subset of features that minimizes redundancy between features, allowing for the features for be more information dense, while maximizing relevance, selecting features that relate to the goal of discriminating circular RNA from other lncRNA. They started with the same 188 features as PredcircRNA, and used mRMR to create subsets of 133 best features. To find the optimal subset among those created, they trained an H-ELM model on each subset and chose the subset that created the model with the highest performance. H-ELM models are very fast to train so repeating the training process for each data subset was feasible without notable time or hardware constraints.

The 10 fold cross validation that took PredcircRNA's MKL model 4 days to complete took H-ELM only 6 hours. This speedup of over 20 times means that models can be trained and evaluated quickly on more modest computer hardware, and faster experimentation can take place.

They also test different models and different numbers of features. MKL with all 188 features achieved 0.801 accuracy, H-ELM with 133 features achieved 0.789, and random forests with 31 features achieved 0.769. The small loss in accuracy with smaller subsets of the data shows that the data is highly redundant and only some features have high relevance, hence mRMR was able to create very small subsets of the features that performed very well.

2.2 Sequence based classification

Related work in natural language processing has also been of great benefit for DNA sequence modeling. Just as a word is made of several letters, a DNA sequence is composed of four different characters. Therefore, we can regard a DNA sequence as a word with letters composed of characters A, C, G, and T. In the following, we present background material related to sentence classification that is relevant to the methodology used in this research project.

2.2.1 GloVe: Global Vectors for Word Representation

Global Vector for Word Representation (GloVe) [15] tackles the same issue that the skip-gram algorithm tackles but from a different angle . Pennington, Socher, and Manning demonstrate that the ratio of co occurrence probabilities of two distinct words includes linguistic data. The concept is close to TF-IDF however it is focused on weighing the significance of a context word during the training of word embeddings.

Their algorithm functions by obtaining word co-occurrence data in the form of the word co occurrence matrix X. Every distinct element, X_{ij}, of a matrix shows the frequency of each word, i, with reference to another word, j. As co-occurrence frequency is able to be instantly encoded in a word-context co-occurrence matrix, GloVe takes this matrix instead of the complete dataset as input. Therefore, we examine our dataset with the following method: for each instance we look for context instances within some section segregated by window size before and after the term. Subsequently the word embeddings are explained in terms of this co-occurrence matrix:

$$w_i^T w_j + b_i + b_j = log(X_{ij}) \tag{2.1}$$

The inventors of GloVe show that the ratio of the co-occurrence probabilities of two words (as opposed to their co-occurrence probabilities themselves) is what holds the information. Therefore they look to encode this data as vector differences. That's why they present a weighted least squares objective J that immediately tries to decrease the variance between the dot product of the vectors of two words and the logarithm of their number of co occurrences.

$$J = \sum_{i=1}^{V} \sum_{j=1}^{V} f(X_{ij})(w_i^T w_j + b_i + b_j - logX_{ij})^2 \tag{2.2}$$

where w_i and b_i are the word vector and bias respectively of word i, w_j and b_j are the same for the word j, X_{ij} is the quantity of occurrence of the word i in the context of the word j. The cost function, f is a weighting function that omits low co-occurrences, which are normally noisy, and also steers clear of overweighting high co-occurrences. The writers fixed x_{max} to 100 and discovered that the hyperparameter $\alpha = \frac{3}{4}$ led to the best empirical results.

$$f(X_{ij}) = \begin{cases} \left(\frac{x_y}{x_{max}}\right)^{\alpha} & if X_{ij} < XMAX \\ 1 & otherwise \end{cases} \tag{2.3}$$

2.2.2 Recurrent Neural Network

A recurrent neural network (RNN) takes a sequence of vectors as input and outputs another sequence of vectors. A very simple RNN is shown in Figure 2.2. Each vector in the sequence, x_t, is multiplied by the weight matrix W_t. The previous hidden state, H_{t-1}, is multiplied by W_w. These two products are added together along with the bias b to get H_t, the hidden state. H_t is used to calculate Y_t, the output at that time step, and the next hidden state.

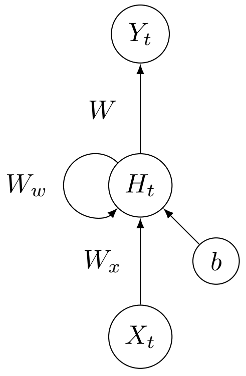

Figure 2.2: A simple RNN

$$H_t = tanh(W_w H_{t-1} + W_x X_t + b) \tag{2.4}$$

The network could translate a sequence of features to a sequence of higher level features, such as converting a sequence of words into predictions for their part of speech. If the output needs to be a fixed-length vector instead of a sequence, only the final hidden state or output could be used.

RNN's are trained by unrolling through time, where the network is treated as a very deep network with a layer at each time step, all sharing the same weights, shown in Figure 2.3. Beause of this, they suffer from the exploding gradient problem (or, conversely, the vanishing gradient problem) where the values of the gradient increase to enormous values. Using a long-short term memory network (see Section 3.1.3.4 for more details on LSTM) can alleviate this problem and provide means for the network to learn long term dependencies.

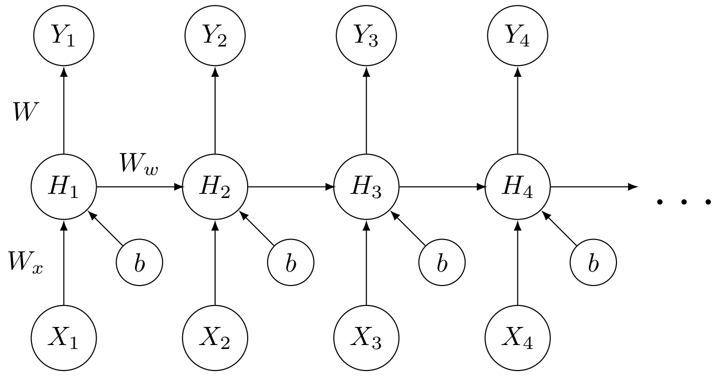

Figure 2.3: The network shown in 2.2 unrolled into a deep feedforward net

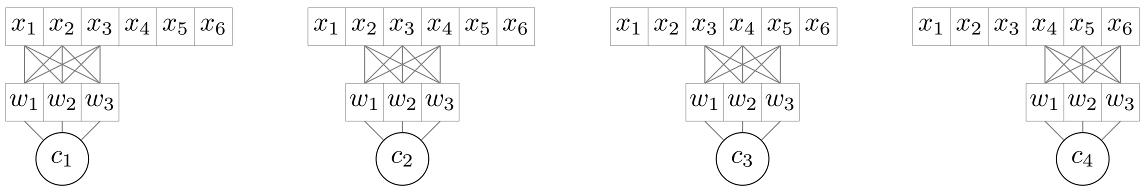

Figure 2.4: The sequence of features $c_{[1...4]}$ created by convolving w over the input $x_{[1...6]}$.The maximum of these would be the value for this feature, $\hat{c}$.

2.2.3 Convolutional Neural Network

Convolutional neural networks are used most commonly in models that work on images, but they have also found use in processing sequences of discrete symbols, like nucleotides in an RNA sequence. They use convolutional filters (Figure 2.4) that take in local features to produce higher level features. Convolutional neural networks can convert a variable length input to a fixed length vector using max pooling, or they can scale the input up or down in size.

Let $x \in \mathbb{R}^k$ be a k-wide vector. A sequence of length n is converted into a rectangular matrix of size (n, k).

$$x_{0:n} = x_1 \otimes x_2 \otimes \ldots \otimes x_{n-1} \otimes x_n \tag{2.5}$$

where $\otimes$ is the concatenation operator.

A convolutional filter $w \in \mathbb{R}^{hk}$ is used a sliding window that covers h words and creates an output feature. Feature c_i is calculated by

$$c_i = f(w \cdot x_{i:i+h1} + b) \tag{2.6}$$

where b is the bias vector, w is the filter, and $x_{i:i+h1}$ is a slice of x the same size as w. f is the activation function, such as tanh, sigmoid, or relu. The filter is scanned across x by incrementing i until the end of the input is reached, creating the output feature c

$$c = [c_1, c_2, \ldots, c_{n-h+1}] \tag{2.7}$$

Max pooling takes the highest value to get a scalar value for $\hat{c}$.

$$\hat{c} = max(c) \tag{2.8}$$

This way a scalar value is created for each filter. All of these features taken together make up a fixed size vector representation of the sequence, which can be used as the input for a fully connected layer.

CHAPTER 3

METHODS

In this section, we describe our multi-modal deep learning framework that integrates different sources of data to predict circular RNAs. We describe how to extract different representations of different sources of which are subsequently integrated using multi-modal deep learning to predict circular RNAs.

3.1 Features extraction

3.1.1 RCM descriptor

Many studies from reading support the speculation that reverse complement matching in flanking introns and circularization are immensely associated. The writers demonstrated in [16] that Reverse complementary sequences between introns bracketing circular RNAs were notably elevated in comparison to linear controls. It has also been verified that the presence of long inverted repeats (IR) flanking the mouse *Sty* genes results in the creation of the *Sty* circular transcript in cultured cells [17]. In [18] the writers provide several lines of evidence to support the deduction that circular RNA formation is reliant on flanking complementary sequences, either with repetitive or nonrepetitive components. Consequently, it has been suggested that RCMs (Figure 3.1) encourage hairpin creation of the transcript which describes how an upstream acceptor site (3' splice site) of an upstream intron is fastened to a downstream donor site (5' splice site) of a flanking downstream intron. A feature like this can be a strong key feature to help our model predict the potential of a sequence to be transformed to circular RNA.

Therefore, we come up first by score H that computes the presence of reverse complementary sequences in flanking sequences. It represents the absolute number of all reverse complement sequences in the flanking sequences. Therefore, for each query sequence S, we take two flanking sequences, each of length L0 base pairs (bps). We split two sequences into k-mers using the sliding window approach. We extract all subsequences of length K with stride $s = 1$, resulting in L subsequences $L = \lfloor L0 - k \rfloor + 1$. There are $D = 4^k$ possible words of length k in the sequences.

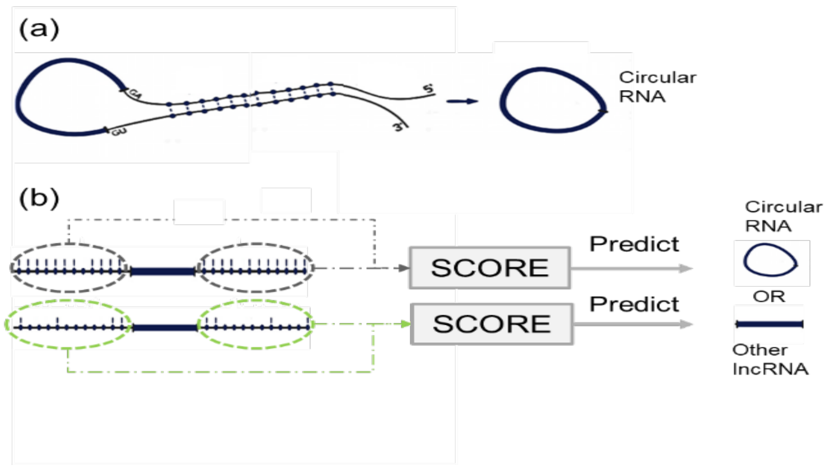

Figure 3.1: (a) Reverse Complement Matches between flanking sequences promote circlarization of RNA. (b) Computing the strength of hairpin in flanking sequences allows circular RNA prediction.

For each word $W_i; i = i, \ldots, 4^k$, we count the number of occurrences in the left flanking sequence $occ_L(W_i)$ and the number of occurrences of the reverse complement to W_i in the right flanking sequence $occ_R(RC(W_i))$.

Then the score of the sequence S is given by eq.3.1 which is the sum of scores H_i of all words W_i where W_i is the absolute number of occurrences of the word W_i:

$$H_{(K,L0)} = \sum_1^{4^k} H_i \tag{3.1}$$

Where $H_i = min(occ_L(W_i), occ_R(RC(W_i)))$

Our score $H_{(k,L0)}$ can be informative about the absolute number of reverse complement sequences. This can be very useful according to current models for the production of circular RNAs, which posit that RNA secondary structures formed by inverted sequences in flanking introns are necessary elements for circularization.

Although it has been suggested that, at minimum, there are 400 corresponding nucleotides required for textitSry circularization *in vivo* [19]. Other research [20] has demonstrated that artificially surrounding an exon with introns containing 800 nucleotides (nt) of flawlessly corresponding duplicates allows for circularization. This means that long reverse complement sequences promotes circularization process. Lacking detailed mechanistic models for how circularization generally makes the prediction of circular RNAs difficult but, based on some previous studies, we will make hypothesis that strengthening the hairpin between the reverse complement sequences may increase the potentiality of circularization. Therefore, maybe the availability of good score that describes the longest reverse complementary sequence in flanking sequences to a query sequence for our classifier may help discrimination between circular RNAs and other lncRNAs.

In the following pseudocode, we will describe our procedure for computing score V that describes the strengthening of the hairpin between flanking sequences. In the results section, we show the effectiveness of such feature for discriminating circular RNAs from other lncRNAs.

In order to evaluate the strengthening of the hairpin between flanking sequences (Figure 3.1b), our method provides a new dynamic programming approach that computes score V for the longest reverse complement sequences while allowing some mismatches and penalizing non-complementary nucleotides. To calculate the longest reverse complement matching between two sequences of length L_1 and L_2 respectively, we split each sequence into overlapping k-mers of length k with stride equal to 1, so now each sequence is represented by $W_i, i = \{1, \ldots, L_1 - k + 1\}$ and $W_j, j = \{1, \ldots, L_2 - k + 1\}$ respectively.

Let $V_{(L1,L2,k)}$ be the score for the longest reverse complementary matching between sequences $W_i, i = \{1, \ldots, L_1 - k + 1\}$ and $W_j, j = \{1, \ldots, L_2 - k + 1\}$

Time complexity of our algorithm to compute $V_{(K,L_1,L_2)}$ is $O(L_1 \times L_2)$. We use k-mers for computing score V. As a constraint, at least 1 k-mer is matched between two flanking sequences, but this algorithm can be extended to eliminate this constraint by simply taking k=1.

3.1.2 Conservation descriptor

The PhastCons [3] method is used to give a score ranging from 0 to 1 to each nucleotide based on its conservation level. We gather pre-computed conservation scores from the UCSC database (https://genome.ucsc.edu). For every exon sequence in every transcript, we average the scores in the exon sequence and then compute the maximum, the average, and the median of those averaged scores. If a transcript doesn't have an exon, the entire transcript is held as a single exon for our computation.

Circular RNAs have a multitude of conserved docking sites for miRNA such as ciRS-7 which holds over 70 selectively conserved miRNA target sites [21]. This is a result of most circular RNAs having almost identical motif sequences. Consequently, we sum the number of frequencies of successive bases whose scores are larger than the specified threshold and then divide the frequency by the whole length of the sequence. We alter the number of successive bases in the scope of 4 through 7,

Algorithm 3.1 Caclculation of score $V_{(k,L_1,L_2)}$

```
for $i = 1, \ldots, L_1 - k + 1$ do
  $v_{(i,0)} = 0$
end for
for $j = 1, \ldots, L_2 - k + 1$ do
  $v_{(0,j)} = 0$
end for
for $i = 1, \ldots, L_1 - k + 1$ do
  for $j = 1, \ldots, L_2 - k + 1$ do
    if $W_i = RC(W_j)$ then
      $v_{(i,j)} = max(0, v_{(i-1,j-1)} + matchscore)$
    else
      $v_{(i,j)} = max(0, v_{(i-1,j-1)} + mismatchscore)$
    end if
  end for
end for
$V_{(k,L_1,L_2)} = max(v_{(i,j)})$ where $i = \{1, \ldots, L_1 - k + 1\}, j = \{1, \ldots, L_2 - k + 1\}$
```

incrementing by 1 and threshold in the scope of 0.5 through 0.7, incrementing by 0.1. This gives us totally 15 features to create our conservation descriptor.

3.1.3 ACNN-BLSTM sequence descriptor

In order to extract ACNN-BLSTM sequence descriptor, we build end-to-end deep learning architecture as shown in Figure 3.2 which consists of three main components: embedding layer, Asymmetric convolutional neural network (ACNN) and Bidirectional long short-term memory network (BLSTM).To begin, we build the general network architecture that we use. We then discuss in depth, k-mer embedding, asymmetric convolutions, and Bidirectional LSTM. K-mer embedding utilizes unsupervised learning to attempt to encode the co-occurrence information of k-mers into low-dimensional vector space. Asymmetric convolutions were utilized to find sequence patterns or motifs. The Bidirectional LSTM were utilized to find far reaching dependencies and form set-length features from varying-length DNA sequences.

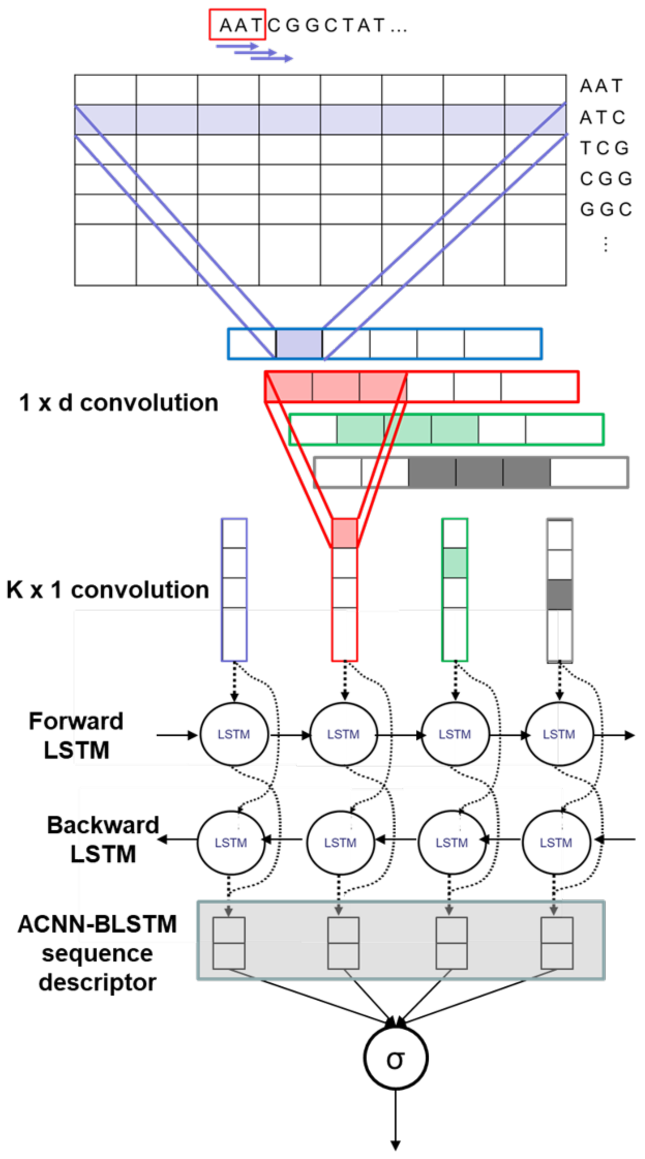

Figure 3.2: Graphical illustration of our end-to-end deep learning architecture used to extract ACNN-BLSTM sequence descriptor.

3.1.3.1 General network architecture

Given a DNA sequence of L_0 base pairs (bps), the first step is to split it into overlapping k-mers of length k using the sliding window approach with stride s, which results in a k-mer sequence with length $L = \lfloor (L_0 - k)/s \rfloor + 1$. Each k-mer is indexed by positive integer in range $C = [1, 2, \ldots, 4^k]$. We will see how we make feature learning for such sequence data $x \in C^L$ with variable length L, i.e. how to learn a feature map $g : C^L \Rightarrow R^d$ that maps $x \in C^L$ into a vector of features $\boldsymbol{h} \in R^d$ helpful for machine learning tasks.

Given N variable-length DNA sequences labeled 0 or 1; 1 stands for circular RNA and 0 for other lncRNA. So, we have N labeled instances $\{x_i, y_i\}_{i=1}^{N}$, where $x_i \in C^L, y_i \in \{0, 1\}$. We aim to learn a function that can assign the label to each instance x_i. Thus, We propose the network shown in Fig. 3.2 in which we decompose the feature learning function $g : C^L \Rightarrow R^d$ into three steps:

$$\boldsymbol{h} = g(\boldsymbol{x}) = g_{lstm}(g_{conv}(g_{embed}(\boldsymbol{x}))). \tag{3.2}$$

The embedding layer calculates the co-occurrence statistics of the k-mers, and learns to represent them in a D-dimensional space, R^D. The convolution stage scours the embedding representation of sequences using a group of convolution filters to encapsulate sequence patterns or motifs. The BLSTM stage performs a Bidirectional LSTM network on the input data to comprehend long-term dependencies, and ultimately produces a constant-length feature vector in R^D.

We process the binary classification as a logistic regression on the feature representations. The conditional likelihood of y_i given $\boldsymbol{x}_i$ and model parameters Θ can be written as:

$$logp(y_i \boldsymbol{x}_i, \boldsymbol{\Theta}) = y_i log\sigma(\boldsymbol{\beta}^T \boldsymbol{h}_i) + (1 y_i log(1 \sigma(\boldsymbol{\beta}^T \boldsymbol{h}_i))), \tag{3.3}$$

where $\boldsymbol{\beta}\ in \mathbb{R}^d$ are the prediction parameters, $\boldsymbol{h}_i \in \mathbb{R}^d$ is the learned fixed-length feature for $\boldsymbol{x}_i$, and $\sigma(z) = 1/(1 + exp(z))$ is the logistic sigmoid function. We train our deep neural network by minimizing the following loss function:

$$l = -\sum_{N}^{i=1} logp(y_i|\boldsymbol{x}_i, \Theta). \tag{3.4}$$

3.1.3.2 k-mer embedding with GloVe

The following describes the process and reason we embed k-mers into a low-dimensional vector space. As previously stated, standard kmer-based methods only compute the vector of k-mer frequencies while not taking into account the co-existence relationships of k-mers. The final k-mer is an unordered document that can be likened to a Bag-of-Words feature [22], which is also commonly utilized in information retrieval and natural language processing. At the same time however, the co-occurence matrix holds global statistical information, which has the possibility of assisting us in the creation of superior feature portrayal. To test this we apply the GloVe model for k-mer embedding. Used for word depiction based on partitioning a matrix of work co-occurrence data, GloVe is superior to other methods that learn vector space portrayals. This is due to the fact that it amalgamates the benefits of the global matrix partitioning and the benefits of the local context window methods.

The statistics of k-mer occurences is the chief source of information obtainable for learning embedding representations. We will designate $\boldsymbol{X}$ to represent the matrix of kmer-kmer co-occurence counts. Here an entry $\boldsymbol{X}_{ij}$ tabulates the number of times that k-mer j occurs in the context window of k-mer i. $i, j \in [1, V]$ are two k-mer indexes, where $V = 4^k$ is the vocabulary size. According to the GloVe model, the cost function to be minimized is,

$$J = \sum_{\substack{i,j=1 \\ \boldsymbol{X}_{ij} \neq 0}}^{V} f(\boldsymbol{X}_{ij})(\boldsymbol{w}_i^T \tilde{\boldsymbol{w}}_j + b_i + \tilde{b}_j - log\boldsymbol{X}_{ij})^2$$

(3.5)

where $\boldsymbol{w} \in \mathbb{R}^D$ are expected k-mer vectors, $\tilde{\boldsymbol{w}} \in \mathbb{R}^D$ are separate context vectors for auxiliary purpose, and $v, \tilde{b} \in \mathbb{R}$ are biases. We can see from the non-decreasing weighting function f in eq.2.3 that the computational complexity depends on the number of nonzero entries in the co-occurrence matrix X. We minimize the cost function in eq. 2.3 using AdaGrad [23] to obtain our embedding vector representations $\boldsymbol{w}_1, \boldsymbol{w}_2, \ldots, \boldsymbol{w}_V \in \mathbb{R}^D$ for all k-mers.

Given these vectors, we can fulfill the embedding stage of feature learning $g_{embed} : C^L \Rightarrow R^{D \times L}$ by embedding every k-mer into the vector space R^D:

$$g_{embed} = [\boldsymbol{w}_1, \boldsymbol{w}_2, \ldots, \boldsymbol{w}_{x_L}], \tag{3.6}$$

where $\boldsymbol{x} = [x_1, x_2, \ldots, x_L] \in C^L$. Based on the output $D \times L$ matrix, we proceed with the convolution stage, which further extracts spatial features using convolutional layers and max-pooling layers.

3.1.3.3 Asymmetric convolutional neural network

As the convolution layer in our model requires fixed-length input, we pad each sequence that has a length less than maxlen with special symbols at the end that indicate the unknown k-mers and for those that have a length longer than maxlen, we simply cut extra k-mers at the end of these sentences to reach maxlen. So now, each sequence is represented as:

$$x_{1:maxlen} = [x_1, x_2, \ldots, x_{maxlen}] \tag{3.7}$$

where $x_j \in R^d$ is the d-dimensional word vector corresponding to the j-th k-mer in the sequence.

A deeper network will generally have more representational power than a shallower network, but the training time and great number of parameters makes them difficult to train. The number of parameters can be greatly reduced with a minor loss in performance by the use of asymmetric convolutions [24], allowing for deeper models to be trained using the same resources.

We divide the more common $k \times d$ rectangular convolutional filter into two separate steps. First we apply $n1 \times d$ convolutions, followed by convolutions [24]. The $1 \times d$ convolution filter is applied to each k-mer x_j in the sentence and generates corresponding feature m_j

$$m_j = f(w^1 \circ x_j + b) \tag{3.8}$$

where $\circ$ is element-wise multiplication, b is the bias and f is a non-linear function. We chose the ReLU activation function, because it is known to perform well in CNNs. We get the feature map $\boldsymbol{m} \in R^L$.

$$\boldsymbol{m} = [m_1, m_2, \ldots, m_L] \tag{3.9}$$

The $k \times 1$ convolution with filter $\boldsymbol{w^2} \in \mathbb{R}^k$ is applied to a window of k features in the feature map $\boldsymbol{m}$ to produce the new feature $\boldsymbol{c_j}$ and the feature map $\boldsymbol{c}$

$$\boldsymbol{c_j} = f(\boldsymbol{w^2} \circ \boldsymbol{m_{j:j+k-1}} + b) \tag{3.10}$$

$$\boldsymbol{c} = [\boldsymbol{c_1}, \boldsymbol{c_2}, \ldots, \boldsymbol{c_{L-k+1}}] \tag{3.11}$$

Dropout is employed Srivastava, et al., 2014 to reduce overfitting. The output of the ACNN is read by a BLSTM, enabling long term dependencies to be captured in both directions.

3.1.3.4 Bidirectional LSTM

Recurrent networks are particularly well suited for finding dependencies and complex relationships in sequential data. However, they can only recognize very short-term dependencies. This problem is overcome by the use of forget gates, which allow some information to be preserved for long stretches of time while selectively forgetting data that is not needed. In this task in particular, dependencies in both directions are important, so a BLSTM network is used to capture this.

An RNN is very much like a single feedforward network, but it preserves a hidden state vector, $h_t \in \mathbb{R}^d$, between time steps. At each time step t, the hidden state vector h_t is a function of the input vector x_t received at time t and its previous hidden state h_{t1}. This transition function can be any differentiable function. Here we use the hyperbolic tangent function.

$$h_t = tanh(Wx_t + Uh_{t-1} + b) \tag{3.12}$$

However, backpropagation through an RNN can lead to an exploding gradient or vanishing gradient [25] [26]. LSTM cells remedy this using a "forget gate" which allows only selective parts of the state through at each time step. This way, values important for long term connections can be maintained and then forgotten when they aren't needed. Equations 3.13 to 3.18 describe the transition equations for an LSTM cell.

We define the LSTM unit at each time step t to be a collection of vectors in $\mathbb{R}^d$: an input gate i_t, a forget gate f_t, an output gate o_t, an input modulation gate g_t, a memory cell c_t and a hidden state h_t. The entries of the gating vectors i_t, f_t and o_t are in $[0, 1]$. The LSTM transition equations are the following:

$$i_t = \sigma(W^{(i)}x_t + U^{(i)}h_{t-1} + b^{(i)}), \tag{3.13}$$

$$f_t = \sigma(W^{(f)}x_t + U^{(f)}h_{t-1} + b^{(f)}), \tag{3.14}$$

$$o_t = \sigma(W^{(o)}x_t + U^{(o)}h_{t-1} + b^{(o)}), \tag{3.15}$$

$$o_t = tanh(W^{(g)}x_t + U^{(g)}h_{t-1} + b^{(g)}), \tag{3.16}$$

$$c_t = i_t \odot g_t + f_t \odot c_{t-1}, \tag{3.17}$$

$$h_t = o_t \odot tanh(c_t), \tag{3.18}$$

t $\mathbb{R}^d$ i_t is the input, f_t is the forget gate, o_t is the output, g_t regulates its input, c_t stores the memory for each time step, and h_t is the hidden state. i_t, f_t and o_t take in real numbers in the range $[0, 1]$. x_t denotes the input from the previous layer, σ denotes the logistic sigmoid function and $\odot$ is element-wise multiplication 3.3.

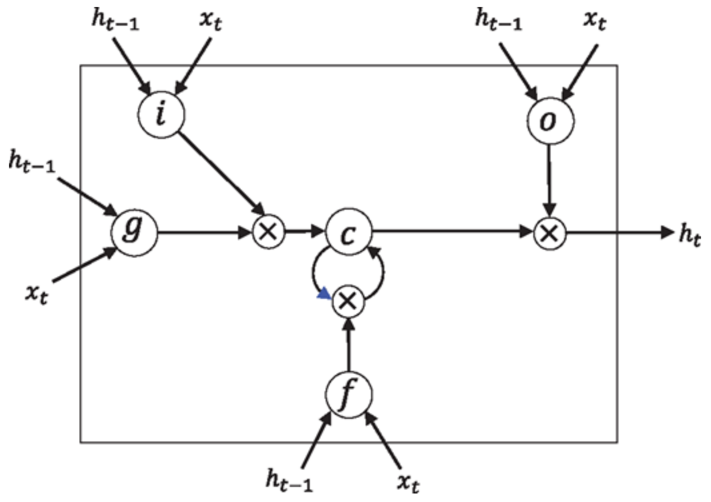

Figure 3.3: Elaborate description of the LSTM unit. i: input gate, f: forget gate, o: output gate, g: input modulation gate, c: memory cell, h: hidden state. The blue arrowhead refers to c_{t1}, namely the memory cell at the previous time step. The notations correspond to Equation 3.13 to 3.18 such that $W^{(\circ)}$ denotes weights for x_t to the output gate, and $U^{(f)}$ denotes weights for h_{t1} to the forget gate, etc. Adapted from [27]

After the LSTM cell has iterated over the sequence, we take its hidden state as output. The hidden state vector is fixed length which makes it suitable for fully connected layers, while also being able to contain information about the entire sequence. The learned representation is limited by the order the vector is read in. It is hard to look for a small detail at the beginning of the sequence that is only relevent until the end. If you read a book that only mentions a character in the first chapter, you would forget what you know about them when they show up in the last chapter. To remedy this with minimal penalty to performance, we run two independent LSTMs over the entire sequence. Dependencies in either direction can be captured, and these passes can be ran in parallel. Their hidden states are concatenated together at the end, resulting in a single fixed length vector with all of the information about the RNA sequence.

3.2 Multimodal learning for circular RNA prediction

Multimodal deep learning is used to learn shared features from different sources of data since we have heterogeneous descriptors from different sources of data. The proposed multi-model in this work can learn and combine the high-level heterogeneous representations simultaneously. the first step in our method is to train the model described in section 2.1.3 and compute the optimal

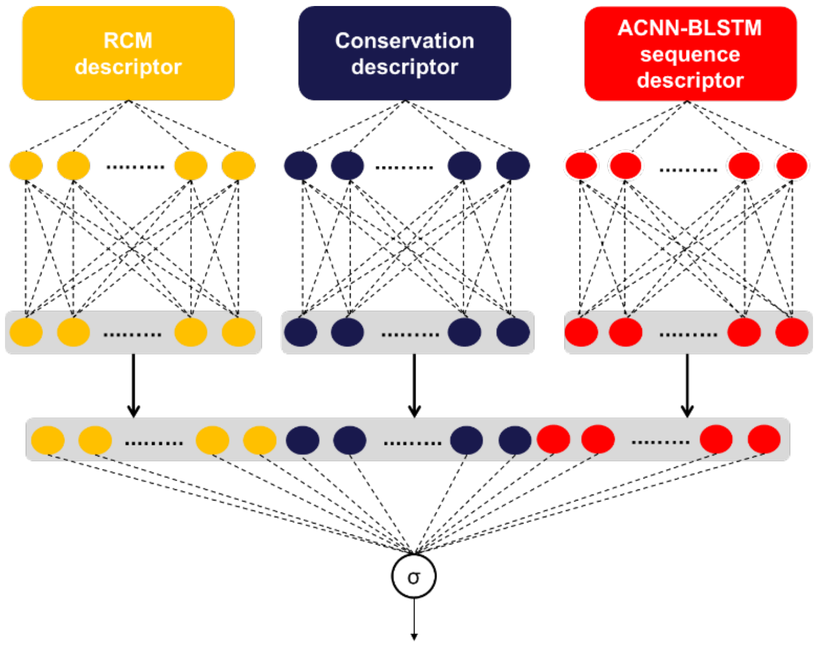

Figure 3.4: The flowchart of proposed circDeep for classification of circular RNA from other lncRNA. It extracts three descriptors and then use multimodal to integrate different representations for classification.

parameters where is the number of hidden layers for ACNN-BLSTM architecture and then for each sequence we compute the ACNN-BLSTM sequence descriptor using the feed-forward pass procedures by taking the final representation before the output layer. We extract at the same time conservation and RCM descriptors using methods described in 3.1.1 and 3.1.2.

The second step of our method is to teach the multi-model with the three obtained descriptors. To achieve this, we can utilize either late fusion or early fusion. Early fusion takes features from the four modalities, chains them together, and then uses the new vector formed to train a deep neural network. The only issue with early fusion is that it fails to consider the different statistical properties of the information it concatenates together. Late fusion creates a final prediction, which is simply an amalgamation of each unimodal prediction from the four modalities. Each unimodal prediction has an output interpreted as a confidence score, these scores are combined to give us the final confidence score. The final prediction allows us to maximize the capabilities of each unimodal classifier, however this method losses some correlation in multi-dimensional space. To avoid this, we use a slight variation of late fusion which is shown in Figure 3.4. In our variation we put each information source through multiple layers in order to construct a high-level representation of each source. The final representation is just an amalgamation of eachindividual high-level representation.

While feature learning, individual DNNs are trained beforehand separately and then fused together for the last common training which utilizes backpropagation. Learned parameters from each model are adjusted automatically during each training epoch. After multiple training epochs, the model can recognize depictions from the RCM, the ACNN-BLSTM sequence, and the conservative descriptors for ensuing categorization. Additionally, the model is better able to learn features for each modality using backpropagation when various modalities are present. This method avoids various problems present in other methods of fusing such as: struggling to recognize highly nonlinear relationships, superfluity and reliance between several descriptors, and over-fitting.

CHAPTER 4

RESULTS AND DISCUSSION

4.1 Experiment setup

In order to gauge the capabilities of our multi-modal deep learning, we used human circular RNAs from the database circRNADb [28] which contains 32,914 human circular RNAs carefully selected from diversified sources. After we removed circular RNAs shorter than 200 nucleotides, we were left with 31,210 circular RNAs to act as our positive dataset. We used GENCODE [29] to create our negative dataset which was made of other lncRNAs such as processed transcript, antisense, lincRNA, sense intronic and sense overlapping. We only chose transcripts that were validated or manually annotated to obtain 19,700 samples that compose our negative data.

We then divide each dataset into training data, validation data, and testing data where 75% is used for training, 10% for validation and 15% for testing. The training data is used to fit the optimal parameters for our model. We then use the validation data to test the performance of the model with these parameters. The test data is then used to test the model with the best performance on the validation data. The testing is used to provide an unbiased evaluation of a final model fit on the training dataset.

For the unsupervised training of k-mer embedding, we generated the corpus of k-mer sequences by setting k to 6, and the strides to 1. Thus, the k-mer vocabulary size is $V=4^6=4096$. Glove is implemented using efficient multicore implementation from https://github.com /stanfordnlp/GloVe. We set the window size (context) to 18, the length of the dense vector to represent each 6-mers to 40, the initial learning rate to 0.025 and the number of iterations to 100.

The circDeep is implemented in python using keras 1.0.4 which is found https://github.com/ fchollet/keras, with the backend of Theano (0.9.0) [30]. In order to benefit from the parallel computation of the tensors, we train our model on a NVIDIA Tesla C2050.

In order to enable the deep architecture to extract ACNN-BLSTM sequence descriptor, we set max length (maxlen) to 8000 bps, the number of asymmetric convolutional filters to 100, the filter length to 4, the memory dimension is set to 100, initial learning rate to 0.02, batch size to 128,

TABLE 4.1

Number of hidden units for two fully connected layers (FCL) for each descriptor.

Layer Output	shape
Embedding_1	(8000, 40)
Dropout_1	(8000, 40)
Convolution1d_1	(7994, 100)
Maxpooling1d_1	(1998, 100)
Dropout_2	(1998, 100)
Convolution1d_2	(1998, 100)
Maxpooling1d_2	(999, 100)
Dropout_3	(999, 200)
Blstm_1	(200)
Dropout_4	(200)
Dense_1	(80)
Dropout_5	(80)
ACNN-BLSTM	(20)
Dropout_6	(20)
Dense_2	(1)

and the maximum number of epochs to 45 (see Table 1 for more details about architecture). This model is optimized using the RMSprop algorithm to learn all the model parameters, including the convolution filters.

When it comes to the multi-modal learning, the number of hidden units for two fully connected layers (FCL) for each descriptor and for the merged representation are listed in Table 2. Batch normalization is added to all hidden layers. We set the initial learning rate set to 0.01, batch size set to 64, the dropout rate set to p=0.3, and the maximum number of epochs to 70. For the measures used for evaluation in our experiments, we used accuracy, sensitivity, specificity, F1 score, and Matthews Correlation Coefficient (MCC).

TABLE 4.2

Number of hidden units for two fully connected layers (FCL) for each descriptor.

# Neurons / Descriptor	First Layer	Second Layer
ACNN-BLSTM sequence descriptor	40	20
Conservation descriptor	128	64
RCM descriptor	512	256

4.2 Efficacy of RCM descriptor

To explore whether the new proposed features $H_{(k,L0)}$ and $V_{(k,L1,L2)}$ are able to predict circular RNAs and improve the performance of our model, we use Pearson's correlation score [28] to measure the relevance of these features and the dependence between the features and the class label (circular RNA or other lncRNA). In other words, Pearson's correlation score is the possibility that $\boldsymbol{H}_{(k,L0)}$ and $\boldsymbol{V}_{(k,L0)}$ features can discriminate between circular RNAs and other lncRNAs. It is one of the most powerful feature selection techniques and it is easy to compute and interpret. The resulting value lies in $[-1;1]$, with -1 meaning perfect negative correlation and +1 meaning perfect positive correlation. The higher score the more relevant the feature is.

We calculate $H_{(k,L0)}$ for K in the interval $[3,11]$ with stride of 1 and $L0$ in the interval $[250,2000]$ with a stride of 250. We calculate also $V_{(k,L1,L2)}$ by taking the same length for flanking sequences $L1 = L2 = L0$ and varying it in the interval $[250,1750]$ with a stride of 250 and k in the interval $[1,7]$ with a stride of 1. We set the matching score to 12 and the mismatch penalty to -2. The results for Pearsons correlation score for our proposed features H and V are shown in Fig.4.1.a and Fig.4.1.b respectively. To make our results more solid, we additionally calculate for the same features Information gain and the results are shown in Fig.4.1.c and Fig.4.1.d .

We can see that the results are very similar and give the same interpretations using any statistical method. Based on the Fig. 4.1 It can be seen that the Pearson's correlation scores for both $H(k,L0)$ and $V(k,L0)$ are considerably high starting from 500 bps in flanking sequences which means that the RCM that promotes the hairpin for circular RNAs are mostly located after 250bps. The results provide preliminary evidence to support our hypothesis that the features $H_{(k,L0)}$ and $V_{(k,L0)}$ for flanking sequences longer than 500bps facilitate the identification of circular RNAs since most Pearsons correlation scores are higher than 0.70 and even reach approximately 0.83.

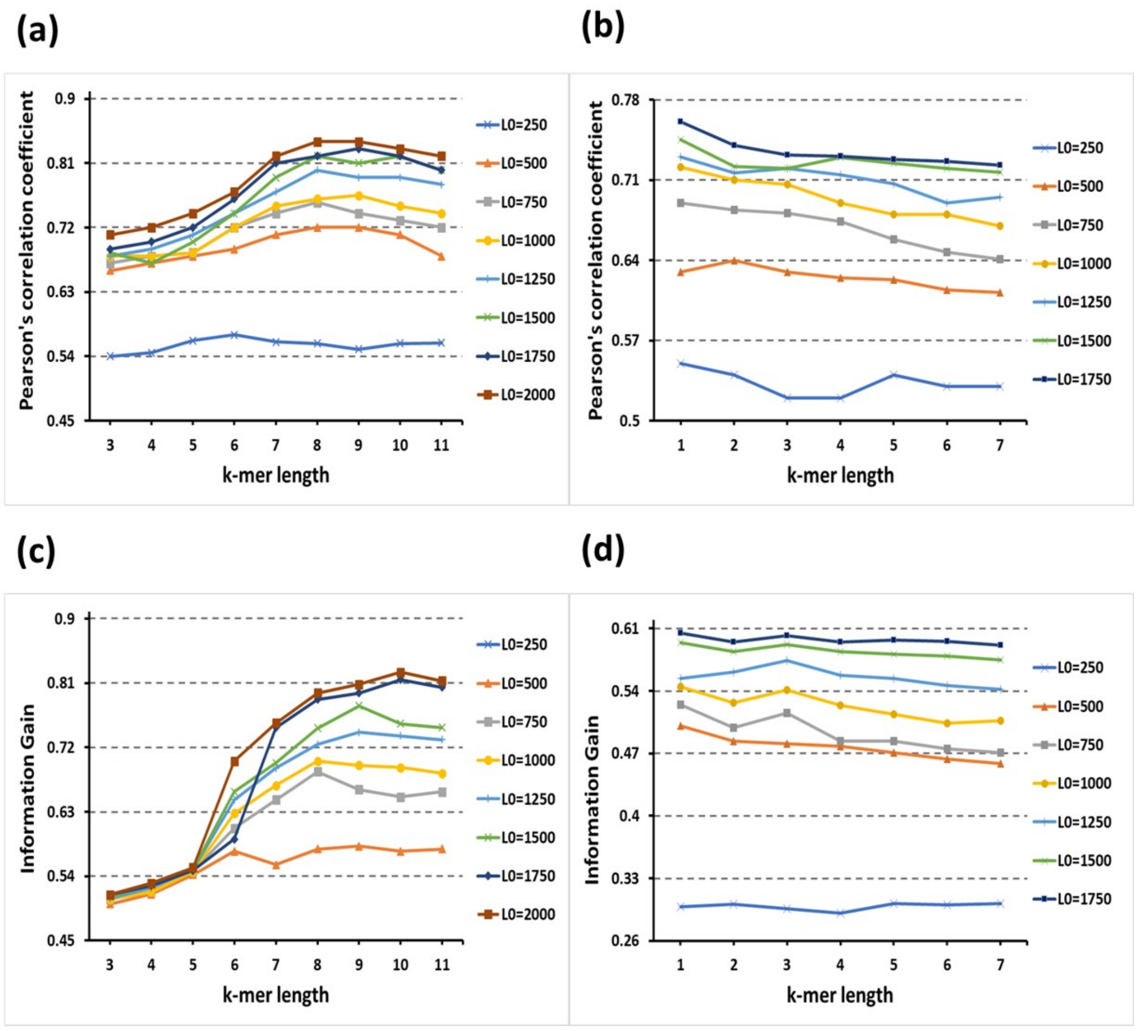

Figure 4.1: (a) Pearson's correlation coefficient as a function of H(k,L0) score for different k-mer length k and different flaking sequence length L0. (b) Pearson's correlation coefficient as a function of V(k,L) score for different k-mer length k and different flaking sequence length L0. (c) Information Gain as a function of H(k,L0) score for different k-mer length k and different flaking sequence length L0. (d) Information Gain as a function of V(k,L) score for different k-mer length k and different flaking sequence length L0.

It can also be seen from fig. 4.1 that Pearson's correlation scores increase when the length of flanking sequences is increased for all values of k-mer length. However, these scores are not extracted for larger k-mer sizes and longer flanking sequences because the run time for both features, $H_{(k,L0)}$ and $V_{(k,L0)}$, is increased with the increased length and size.

When we compare features H and V for any fixed length of flanking sequences from fig. 4.1, $H_{(k,L0)}$ features reached better Pearson's correlations scores and Information Gain scores than $V_{(k,L0)}$ features. This shows that the absolute number of RCM is more informative than the score of the longest hairpin in the flanking sequences when it comes to the possibility of a sequence to be translated to circular RNA. Therefore, our results strongly indicate that circularization can be altered by the competition of RNA pairing across flanking introns and does not rely on just the one longest reverse complement sequence.

Since we aim to obtain a robust classifier, we rank the features using Pearson's correlation score and then we select the first 70 ranked features to obtain the final RCM descriptor composed of 42 $H_{(k,L0)}$ features and 28 $V_{(k,L0)}$ features.

4.3 Efficacity of ACNN-BLSTM sequence descriptor

4.3.1 Comparing with baseline architectures for sequence classification

To evaluate the efficacity of our proposed ACNN-BLSTM sequence descriptor, we compared the performance of our proposed architecture in Table 4.3 with several other methods for text/sequence classification. We compared with a one hot-CNN architecture adopted by [29] which is composed of one-hot coded binary matrix, convolutional layers, global mix-pooling layer, and a fully connected layer; One hot-CNN-BLSTM adopted in [31] composed of one-hot coded binary matrix, convolutional layer followed by rectified linear units, two layers BLSTM and finally a network of fully connected layers; Glove-CNN-BLSTM adopted by [32] composed of k-mer embedding with Glove, convolutional layers, max-pooling layer, followed by a Bidirectional LSTM, and a fully connected layer; One-layer LSTM composed of the same embedding layer as our architecture followed by a one-layer LSTM; One-layer Bi-LSTM composed of the same embedding layer as our architecture followed by a one-layer BLSTM, ACNN architecture which is same as our architecture but without the BLSTM layer, and a CNN that we implemented which is same as the ACNNs architecture with the exception of the convolutional layer being non-asymmetric. Table 4.3 shows the accuracy on training and validation sets and the accuracy and MCC on test set for all methods described above

TABLE 4.3

Performance comparison between our ACNN-BLSTM proposed architecture and other baseline methods for sequence classification

Method	Train ACC	Val ACC	Test ACC	Test MCC	#epochs	Time
One-hot-CNN	0.8176	0.8185	0.8117	0.6164	24	368s
ACNN	0.8402	0.8385	0.8330	0.6625	32	308s
CNN	0.8225	0.8208	0.8194	0.6299	38	338s
LSTM	0.7836	0.7820	0.7826	0.5568	24	712s
BLSTM	0.8029	0.8034	0.8020	0.5997	31	748s
Glove-CNN-BLSTM	0.8804	0.8756	0.8737	0.7324	48	354s
One-hot-CNN-BLSTM	0.8609	0.8570	0.8541	0.6890	42	394s
ACNN-BLSTM	0.8947	0.8935	0.8933	0.7730	36	347s

in addition to the number and average running time per epoch.

By comparing different models, we noticed that convolutional neural networks (one hot-CNN, ACNN, CNN) outperform LSTM related models (LSTM, BLSTM); this validates the importance of convolutional operations in predicting circular RNAs. The ACNN architecture is considered the fastest and the best among convolutional neural networks, it reached 0.833 accuracy and 0.6625 MCC on test data which demonstrates its power for capturing local sequence patterns and detecting spatial motifs that characterize the circular RNAs. For LSTM related models, BLSTM architecture achieved best performance with 0.8020 accuracy and 0.5997 MCC. This demonstrates that it is more useful to have access to both, the past and the future context for predicting circular RNAs.

The most interesting aspect of our results is how the performance is improved by combining convolutional neural network with recurrent neural network This can be best explained by considering the abilities of both recurrent, and convolutional neural networks. Recurrent neural networks excel at sequential modelling while convolutional networks completely fail at learning sequential correlations. Recurrent neural networks fail to derive features in parallel but convolutional networks can learn local responses from dimensional data. Consequently, the amalgamation of the two networks gives us both the benefits from the two and allows us to predict circular RNA more accurately. Our method completely eclipses the baseline models when predicting circular RNA; it achieved 0.8933 accuracy and 0.7730 MCC on test data. We achieved almost identical results when we ran validation and test data, meaning we did not overfit the model by using validation data and an early stop strategy.

4.3.2 Visualization of k-mer embedding

In order to allow our model to be more easily understandable, we continue with a comprehensive analysis of k-mer embedding. Where our method is different than other top tier deep learning methods when it comes to genomic analysis is our usage of k-mer embedding vectors, trained with GloVe, as a portrayal of DNA sequences. GloVe is an unsupervised learning algorithm which trains based on amassed global kmer-kmer co-existence data from a compilation of k-mer sequences. The final portrayal demonstrates linear infrastructures of the vector space, which aid future classification functions. To begin with, we divide circular RNAs sequences into k-mer sequences. We then take each k-mer and treat it as a word and therefore treat each k-mer sequence as a sentence. All together we have a vocabulary of 4096 words or k-mers. The five most prevalent k-mers along with their frequencies are listed as follows, (tttttt, 3183404), (aaaaaa, 2038017), (attttt, 1363303), (tatttt, 1181591), and (ttttta, 1156308). The least prevalent k-mers are as follows (cgcgaa, 2035), (cgcgta, 2299), (tacgcg, 2455), and (cgaacg, 2497).

The distribution of k-mer frequencies is depicted in Fig 4.2.a. Fig 4.2.b depicts the symmetric co-existence matrix, which is normalized using the common logarithm function to clarify the model. Our dataset contains values less than 1.0 because of our application of a function weighed on distance to while calculating the co-existence matrix. There was also a copious amount of zero entries, 20,485, in our data set. These are most often found in the bottom right corner in white.

Instinctually, it can be seen that the co-existence matrix involves complicated trends and an abundance of data on k-mer dependencies on DNA sequences.

In fig 4.2.c we can see the 100-dimensional embedding vectors of k-mers learned by GloVe. The embedding matrix has an average value of 0.000074 with a maximum of 0.009795 and a minimum of -0.009814. The data best fits a Gaussian distribution with a P-value 1.3 e-13. This dimension is beneficial to feature representation, as it is generally decentralized on every dimension. We use two methods to show the dimension reduction response: Principal Component Analysis(PCA) and t-Distributed Stochastic Neighbor Embedding(t-SNE) in Fig 4.2.d. We use colored circles to plot k-mers and a continuous colormap to represent each k-mers frequency. It is important to note that even though the two techniques produce different outcomes, they both show that k-mers with similar

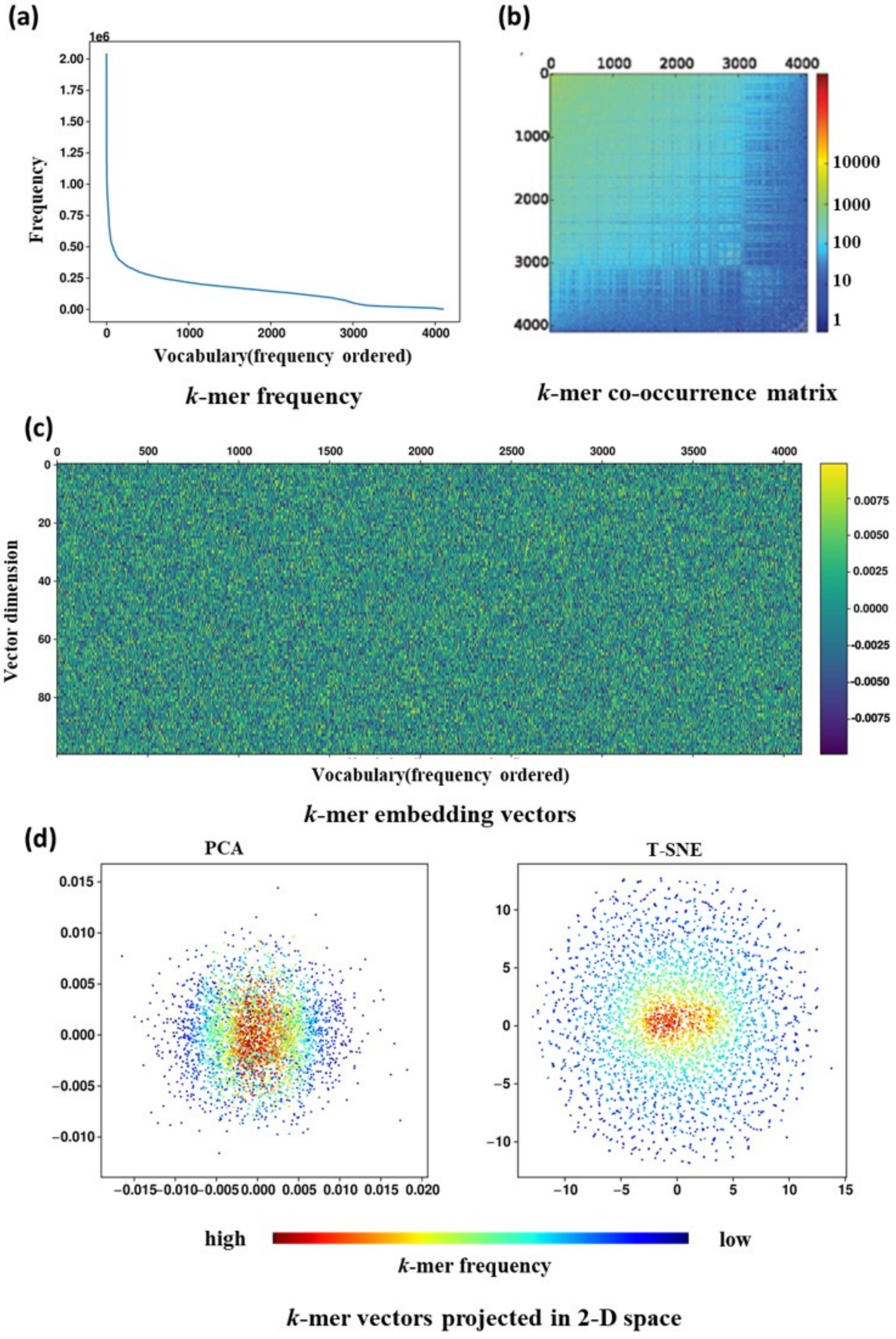

Figure 4.2: Visualization of k-mer statistics and k-mer embedding vectors. (a) the frequency of k-mers. (b) the co-occurrence matrix of k-mers with a log-normalized colormap. (c) the k-mer vectors produced by GloVe. (d) visualization of all the k-mers by projecting their vectors into a plane using PCA and t-SNE

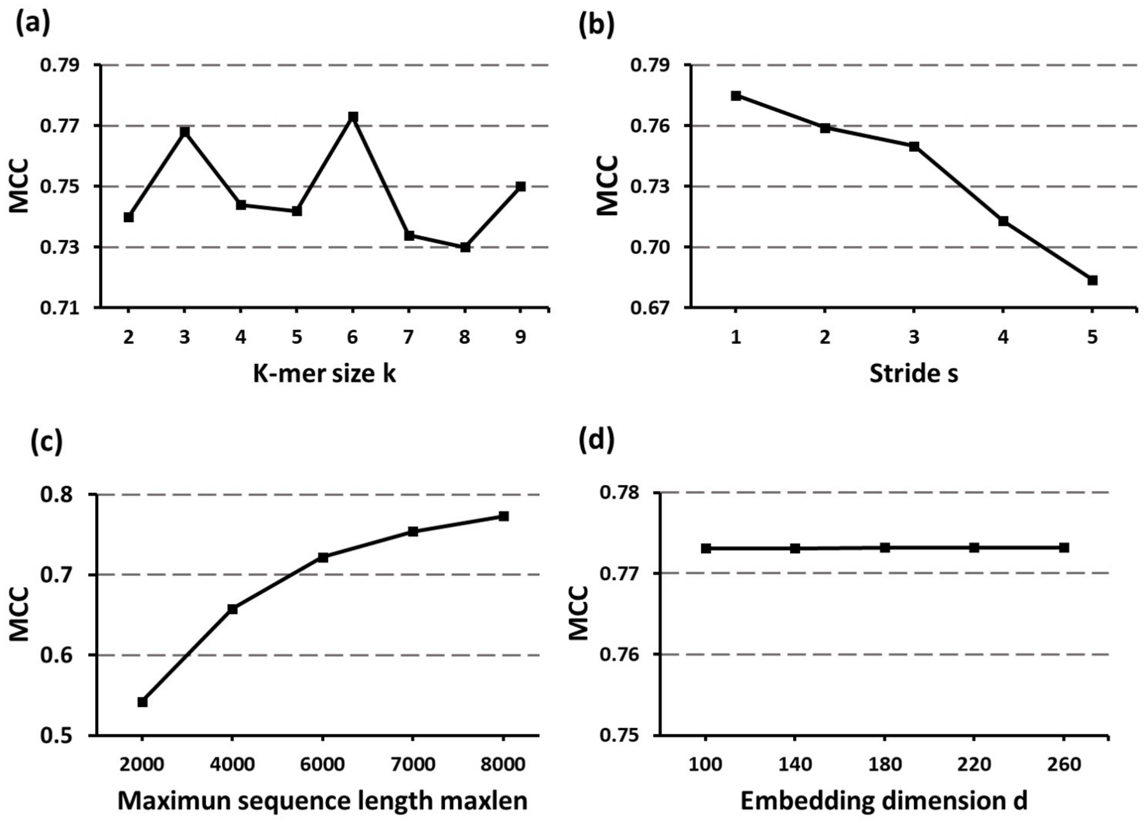

Figure 4.3: Matthews Correlation Coefficient (MCC) scores for test data with varying four hyper-parameters.

frequencies are usually neighboring each other in the embedding vector space.

4.3.3 Model analysis

In order to analyze our architecture used to extract ACNN-BLSTM descriptor, we evaluate its robustness and interpret the influence of four hyper-parameters: the k-mer length k, the splitting stride *s*, the maxlen parameter, and the embedding dimension *d*. We evaluate our model using the MCC measure on the test set, the results are shown in Figure 4.3.

Starting with the k-mer length, *k*, we expected that increasing *k* would increase the total number of different k-mers, increase the information in the co-occurrence statistics, and improve our model. However, we find that k=6 gives the best performance. In fact, The performance is good with any multiple of 3 in the k-mer size. This leads us to the importance of mining the sequences with trinucleotides for circular RNAs prediction. Even though circular RNAs are classified as non-coding RNAs, recently, several studies demonstrate that circular RNAs can be translated [30,31,33].

In agreement with these finding, the peak at the k-mer size of 3, i.e., size of one amid acid, in Figure 4.3a suggests that the translation of circular RNAs is not trivial.

The second hyper-parameter is splitting stride, s. As expected, a larger stride will decrease the corpus size, which in turn, decreases the contained information in the sequence, and decreases the performance. Therefore, we set stride at $s = 1$ in our model.

The third hyper-parameter is maximum sequence length, maxlen. There is a positive correlation between MCC and maxlen. The reason behind the correlation is obvious as some input sequences are truncated at shorter lengths, which results in loss of information contained in sequence. We were limited to a maxlen of 8000bps, due to limitations in computer memory.

The final hyper-parameter is the embedding dimension d. We can see that it has no impact on the performance of the model, however, when d is increased, the time to train the weights of the embedding matrix is increased; This is which leads us to choose $d = 100$ in our model.

Next, we investigate the impact of different filter configurations in the asymmetric convolutional layer on the model performance. We show in Fig 4.4 the MCC measure on the testing data using different filter configurations. For each filter configuration, we report in Figure 4.4 the best result under extensive grid-search on hyper-parameters. In the asymmetric convolutional layer of our model, filters are utilized to represent local 6-mers features. We inherently thought that several convolutional layers in parallel with filters of various sizes would perform better than any single asymmetric convolutional layer with filters of the same length due to different sized filters being able to find features of different n-grams (n consecutive 6-mers). Nevertheless, our experiments demonstrated that a single convolutional layer with a filter length of 4 always surpasses the other options. When multiple layers were run together, filter combinations with a filter of length 4 always performed better. This further aids the verification of 4-gram features playing a vital role in representing local features while predicting circular RNA.

4-grams means we extract local features from every four overlapping 6-mers which means three non-overlapping trinucleotides. This brings us back to the results found in [34] that using the word2vec skip-gram model was able to generate trigram of amino acids representations (1 amino acid is equivalent to trinucleotide in DNA sequence) that reproduced known physical relationships and were useful for protein classification. our results show again how circular RNAs sequences share the characteristics of the coding sequences when it comes to mining the sequences with text mining techniques where we treat the sequence as collection of k-mers.

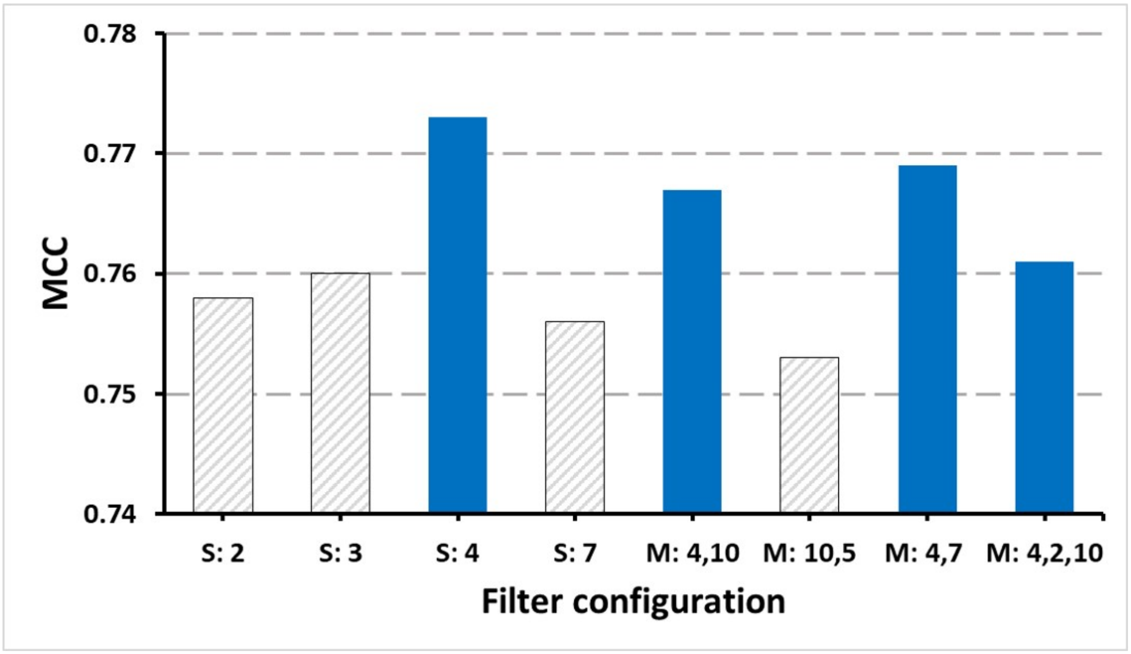

Figure 4.4: Matthews Correlation Coefficient scores for test data with different filter size strategies. For the horizontal axis, S means single convolutional layer with the same filter length, and M means multiple convolutional layers in parallel with different filter lengths. In both single- and multiple-filter lengths, a filter of length 7 always showed higher MMC score (blue bars).

4.4 Multimodal performance for classification of circular RNA from other lncRNA

Table.4.4 shows the performance of unimodal classifiers, bimodal classifiers, multimodal classifiers and PredcircRNA. For unimodal classifiers, we stack two fully connected layers for each descriptor. For bimodal classifiers, we fuse each two descriptors using our multimodal approach. For multimodal classifiers, we fuse all three descriptors using our method, early fusion and late fusion. We compared also the time needed to extract all features on the test data which contains 7630 samples. We didn't report the time needed to test classifiers after extracting features because it is negligible compared to time of extracting features.

As indicated in Table 4.4, ACNN-BLSTM sequence descriptor outperforms two other descriptors with a gap of more than 15% in F1 score which reflects its huge importance for our task. Even though our proposed RCM descriptor has the least performance capability with high running time to be extracted, it is considered good with 0.64 F1 score. We can see also its effect in improving the performance when combined with other descriptors.

It is interesting that only 0.2 minutes is required to extract the ACNN-BLSTM descriptor

TABLE 4.4

Performance comparison between unimodal classifiers, bi-modal classifiers, multimodal approaches and PredcircRNA. SEQ: ACNN-BLSTM sequence descriptor, CONS: conservation descriptor, RCM: RCM descriptor, Time: time needed to extract features used for classification for test data.

Model	Accuracy	MCC	F1 score	Time (minutes)
SEQ	0.8977	0.7792	0.8596	**0.2**
CONS	0.7611	0.4969	0.6897	15
RCM	0.7158	0.4072	0.64	27
SEQ + RCM	0.9271	0.8542	0.9278	47.2
SEQ + CONS	0.9333	0.8654	0.9333	15.2
RCM + CONS	0.8478	0.6838	0.8108	62
Our fusion	**0.9417**	**0.8833**	**0.9402**	62.2
Early fusion	0.9271	0.8542	0.9278	62.2
Late fusion	0.9314	0.8628	0.9307	62.2
PredcircRNA	0.8056	0.6113	0.8108	515

with an 0.8596 F1 score while PredcircRNA required approximately 8 hours and 35 minutes to extract all features, such as graph features, component composition features, including frequencies of trinucleotides, conservation score features, features of ALU and tandem repeats, the ORF, and SNPs from transcripts (GraphProt is taking most of running time) with only an 0.8108 F1. This reflects the importance of sequence patterns and long-range dependencies between 6-mers in predicting circular RNAs.

As expected, Table 4.4 shows that the multimodal fusion produces much higher performance than any individual modality. This indicates the strong complementarity shared between the three descriptors. Furthermore, our multimodal fusion leads to a 0.9402 F1 score. This is greater than all unimodal learners and also other multimodal fusion baseline methods.

To once again make a comparison with PredcircRNA and H-ELM, we tested our tool with the data proposed by [12], and the results are shown in Figure 4.5. This further supported our results and interpretations by having an improvement of more than 12% in accuracy using our proposed method.

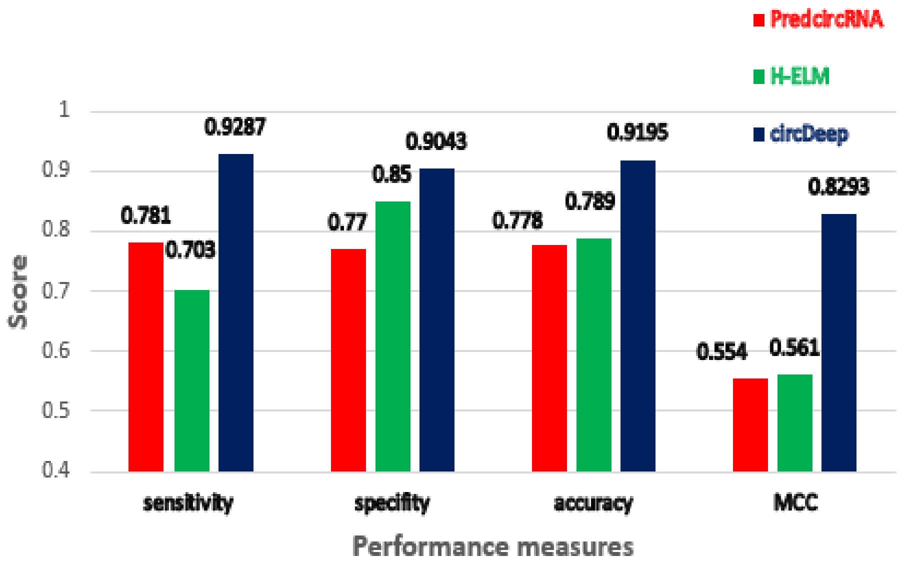

Figure 4.5: Comparison of circular RNA prediction tools: circDeep significantly outperforms previous proposed methods

4.5 Availability

The source code can be downloaded from https://github.com/UofLBioinformatics/circDeep upon request.

CHAPTER 5

CONCLUSIONS AND POTENTIAL FUTURE WORK

In this thesis, we propose a new multimodal deep learning tool to distinguish circular RNA from other lncRNA. Compared to existing approaches, our approach has the following advantages:

1. It takes advantage of our proposed RCM descriptor which can provide the likelihood of circularization given the flanking sequences and the query sequences.

2. It successfully fuse ACNN, BLSTM and DNNs for dealing with the heterogeneous input data types to improve discrimination ability.

3. Our tool achieved an improvement of more than 12% in accuracy in both datasets with a very small running time compared to existing tools.

Recently there have also been a growing number of circular RNAs identified in different species, but for now our model can only detect human circular RNAs. In the future , we plan to train different models and each model for a different species.

REFERENCES

[1] John S. Mattick and Igor V. Makunin, "Non-coding rna," *Human Molecular Genetics*, vol. 15, no. suppl_1, pp. R17–R29, 2006.

[2] Tim R. Mercer, Marcel E. Dinger, and John S. Mattick, "Long non-coding rnas: insights into functions," vol. 10, pp. 155, 2009.

[3] Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean, "Efficient estimation of word representations in vector space," *arXiv preprint arXiv:1301.3781*, 2013.

[4] Jennifer Harrow, Adam Frankish, Jose M Gonzalez, Electra Tapanari, Mark Diekhans, Felix Kokocinski, Bronwen L Aken, Daniel Barrell, Amonida Zadissa, Stephen Searle, et al., "Gencode: the reference human genome annotation for the encode project," *Genome research*, vol. 22, no. 9, pp. 1760–1774, 2012.

[5] Thomas A. Cooper, Lili Wan, and Gideon Dreyfuss, "Rna and disease," *Cell*, vol. 136, no. 4, pp. 777–793, 2009.

[6] Maria Eriksson, W. Ted Brown, Leslie B. Gordon, Michael W. Glynn, Joel Singer, Laura Scott, Michael R. Erdos, Christiane M. Robbins, Tracy Y. Moses, Peter Berglund, Amalia Dutra, Evgenia Pak, Sandra Durkin, Antonei B. Csoka, Michael Boehnke, Thomas W. Glover, and Francis S. Collins, "Recurrent de novo point mutations in lamin a cause hutchinson-gilford progeria syndrome," *Nature*, vol. 423, no. 6937, pp. 293–298, 2003.

[7] Chantal F. Morel, Mary Ann Thomas, Henian Cao, Caroline H. ONeil, J. Geoffrey Pickering, William D. Foulkes, and Robert A. Hegele, "A lmna splicing mutation in two sisters with severe dunnigan-type familial partial lipodystrophy type 2," *The Journal of Clinical Endocrinology & Metabolism*, vol. 91, no. 7, pp. 2689–2695, 2006.

[8] Liang Chen, Chuan Huang, Xiaolin Wang, and Ge Shan, "Circular rnas in eukaryotic cells," *Current Genomics*, vol. 16, no. 5, pp. 312–318, 2015.

[9] Sebastian Memczak, Marvin Jens, Antigoni Elefsinioti, Francesca Torti, Janna Krueger, Agnieszka Rybak, Luisa Maier, Sebastian D. Mackowiak, Lea H. Gregersen, Mathias Munschauer, Alexander Loewer, Ulrike Ziebold, Markus Landthaler, Christine Kocks, Ferdinand le Noble, and Nikolaus Rajewsky, "Circular rnas are a large class of animal rnas with regulatory potency," *Nature*, vol. 495, no. 7441, pp. 333–338, 2013.

[10] Erika Lasda and Roy Parker, "Circular rnas: diversity of form and function," *RNA*, vol. 20, no. 12, pp. 1829–1842, 2014.

[11] Thomas Derrien, Rory Johnson, Giovanni Bussotti, Andrea Tanzer, Sarah Djebali, Hagen Tilgner, Gregory Guernec, David Martin, Angelika Merkel, and David G Knowles, "The gencode v7 catalog of human long noncoding rnas: analysis of their gene structure, evolution, and expression," *Genome research*, vol. 22, no. 9, pp. 1775–1789, 2012.

[12] Xiaoyong Pan and Kai Xiong, "Predcircrna: computational classification of circular rna from other long non-coding rna using hybrid features," *Molecular BioSystems*, vol. 11, no. 8, pp. 2219–2226, 2015.

[13] Xiaoyong Pan and Hong-Bin Shen, "Rna-protein binding motifs mining with a new hybrid deep learning based cross-domain knowledge integration approach," *BMC bioinformatics*, vol. 18, no. 1, pp. 136, 2017.

[14] Lei Chen, Yu-Hang Zhang, Guohua Huang, Xiaoyong Pan, ShaoPeng Wang, Tao Huang, and Yu-Dong Cai, "Discriminating cirrnas from other lncrnas using a hierarchical extreme learning machine (h-elm) algorithm with feature selection," *Molecular Genetics and Genomics*, 2017.

[15] Andrej Karpathy and Li Fei-Fei, "Deep visual-semantic alignments for generating image descriptions," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2015, pp. 3128–3137.

[16] Andranik Ivanov, Sebastian Memczak, Emanuel Wyler, Francesca Torti, Hagit T Porath, Marta R Orejuela, Michael Piechotta, Erez Y Levanon, Markus Landthaler, and Christoph Dieterich, "Analysis of intron sequences reveals hallmarks of circular rna biogenesis in animals," *Cell reports*, vol. 10, no. 2, pp. 170–177, 2015.

[17] Xiao-Ou Zhang, Hai-Bin Wang, Yang Zhang, Xuhua Lu, Ling-Ling Chen, and Li Yang, "Complementary sequence-mediated exon circularization," *Cell*, vol. 159, no. 1, pp. 134–147, 2014.

[18] Meng Huanhuan and Zhang Yue, "Classification of electrocardiogram signals with deep belief networks," in *Computational Science and Engineering (CSE), 2014 IEEE 17th International Conference on*. pp. 7–12, IEEE.

[19] Robert A Dubin, Manija A Kazmi, and Harry Ostrer, "Inverted repeats are necessary for circularization of the mouse testis sry transcript," *Gene*, vol. 167, no. 1, pp. 245–248, 1995.

[20] Thomas B Hansen, Trine I Jensen, Bettina H Clausen, Jesper B Bramsen, Bente Finsen, Christian K Damgaard, and Jrgen Kjems, "Natural rna circles function as efficient microrna sponges," *Nature*, vol. 495, no. 7441, pp. 384–388, 2013.

[21] Yoav Goldberg and Omer Levy, "word2vec explained: deriving mikolov et al.'s negative-sampling word-embedding method," *arXiv preprint arXiv:1402.3722*, 2014.

[22] Zellig S. Harris, "Distributional structure," *¡i¿WORD¡/i¿*, vol. 10, no. 2-3, pp. 146–162, 1954.

[23] John Duchi, Elad Hazan, and Yoram Singer, "Adaptive subgradient methods for online learning and stochastic optimization," Tech. Rep. UCB/EECS-2010-24, EECS Department, University of California, Berkeley, Mar 2010.

[24] C. Szegedy, Liu Wei, Jia Yangqing, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich, "Going deeper with convolutions," in *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 1–9.

[25] Sepp Hochreiter, "The vanishing gradient problem during learning recurrent neural nets and problem solutions," *International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems*, vol. 06, no. 02, pp. 107–116, 1998.

[26] Y. Bengio, P. Simard, and P. Frasconi, "Learning long-term dependencies with gradient descent is difficult," *IEEE Transactions on Neural Networks*, vol. 5, no. 2, pp. 157–166, Mar 1994.

[27] S. Kaae Sønderby, C. Kaae Sønderby, H. Nielsen, and O. Winther, "Convolutional LSTM Networks for Subcellular Localization of Proteins," *ArXiv e-prints*, Mar. 2015.

[28] Mark A Hall, "Correlation-based feature selection of discrete and numeric class machine learning," 2000.

[29] Haoyang Zeng, Matthew D. Edwards, Ge Liu, and David K. Gifford, "Convolutional neural network architectures for predicting dnaprotein binding," *Bioinformatics*, vol. 32, no. 12, pp. i121–i127, 2016.

[30] Ivano Legnini, Gaia Di Timoteo, Francesca Rossi, Mariangela Morlando, Francesca Briganti, Olga Sthandier, Alessandro Fatica, Tiziana Santini, Adrian Andronache, Mark Wade, Pietro Laneve, Nikolaus Rajewsky, and Irene Bozzoni, "Circ-znf609 is a circular rna that can be translated and functions in myogenesis," *Molecular Cell*, vol. 66, no. 1, pp. 22–37.e9, 2017.

[31] Deirdre C. Tatomer and Jeremy E. Wilusz, "An unchartered journey for ribosomes: Circumnavigating circular rnas to produce proteins," *Molecular Cell*, vol. 66, no. 1, pp. 1–2, 2017.

[32] X. Min, W. Zeng, N. Chen, T. Chen, and R. Jiang, "Chromatin accessibility prediction via convolutional long short-term memory networks with k-mer embedding," *Bioinformatics*, vol. 33, no. 14, pp. i92–i101, 2017.

[33] Nagarjuna Reddy Pamudurti, Osnat Bartok, Marvin Jens, Reut Ashwal-Fluss, Christin Stottmeister, Larissa Ruhe, Mor Hanan, Emanuel Wyler, Daniel Perez-Hernandez, Evelyn Ramberger, Shlomo Shenzis, Moshe Samson, Gunnar Dittmar, Markus Landthaler, Marina Chekulaeva, Nikolaus Rajewsky, and Sebastian Kadener, "Translation of circrnas," *Molecular Cell*, vol. 66, no. 1, pp. 9–21.e7, 2017.

[34] Ehsaneddin Asgari and Mohammad RK Mofrad, "Continuous distributed representation of biological sequences for deep proteomics and genomics," *PloS one*, vol. 10, no. 11, pp. e0141287, 2015.

CURRICULUM VITAE

NAME: Mohamed Chaabane

ADDRESS: Computer Engineering & Computer Science Department

Speed School of Engineering

University of Louisville

Louisville, KY 40292

EDUCATION:

M.Sc., Computer Science & Engineering

May 2018

University of Louisville, *Louisville, Kentucky*

B.Eng., Computer Science

June 2016

Tunisia Polytechnic School, *Tunis, Tunisia*

PROJECTS AND INTERNSHIPS:

1. circDeep: End-to-end learning framework for circular RNA classification from other long non-coding RNAs using multi-modal deep learning (https://github.com/UofLBioinformatics/circDeep)

2. B.Eng graduation internship Project, Performed at Multimedia Research Lab from March 2016 to August 2016. In this project, we developed activity based scene retrieval framework to detect and classify scenes that involve Cardio-Pulmonary Resuscitation (CPR) activity from training video sessions simulating medical crises.

PUBLICATIONS:

1. **M. Chaabane**, E.C. Rouchka, and J.W. Park, *" Circular RNA Detection from High-throughput Sequencing"*, Proceedings of the International Conference on Research in Adaptive and Convergent Systems. 2017, ACM: Krakow, Poland. p. 19-24.