Application of Deep Learning Models to MicroRNA Transcription Start Site Identification

Clayton Barham, Mingyu Cha, Xiaoman Li, Haiyan Hu
Department of Computer Science
University of Central Florida
Orlando, United States of America
e-mail: haihu@cs.ucf.edu

***Abstract*—MicroRNAs (miRNA) are ~22 base pair long RNAs that play important roles in regulating gene expression. Understanding the transcriptional regulation of miRNA is critical to gene regulation. However, it is often difficult to precisely identify miRNA transcription start sites (TSSs) due to miRNA-specific biogenesis. Existing computational methods cannot effectively predict miRNA TSSs. Here, we employed deep learning architectures incorporating Long Short-Term Memory (LSTM) and Convolutional Neural Network (CNN) techniques to detect miRNA TSSs in regions of accessible chromatin. By testing on benchmark experimental data, we demonstrated that deep learning models outperform support vector machine and can accurately distinguish miRNA TSSs from both flanking regions and intergenic regions.**

Keywords-bioinformatics; transcriptional regulation; transcription start sites; neural networks; deep learning

I. INTRODUCTION

Micro-RNA (miRNA) refers to a class of non-coding RNA that plays a role in post-transcriptional regulation. miRNA are typically ~22 nucleotides in length and play a role in the down regulation of the expression of more than 30% of mammalian gene products by binding to the corresponding mRNA [1-3]. MiRNAs regulate biological processes such as cell differentiation, development, and apoptosis. Misexpression of miRNAs has been associated with diseases such as diabetes, cancer, and heart disease [4, 5]. Understanding the regulation and expression of miRNAs is an essential component of understanding gene regulation and its role in disease phenotypes.

Transcription Start Sites (TSS) are the locations within a promoter region where the transcription of gene products begins. The TSSs of genes that produce miRNAs are more difficult to study than their counterparts in genes that produce proteins, due to the biogenesis process undergone by miRNAs [6, 7]. Long sequences of primary miRNAs (pri-miRNA) are transcribed from the genes that ultimately produce mature miRNAs. The pri-miRNAs are then processed by nuclear RNase III Drosha and a cofactor protein to produce precursor miRNAs (pre-miRNA). The pre-miRNAs are then cleaved by the RNase III Dicer to produce RNA-induced Silencing Complexes (RISCs). The RISC, together with an AGO protein, is involved to produce the mature miRNAs. The length of the pri-miRNAs relative to the mature miRNAs means that the TSSs for miRNAs can be surprisingly distant from the mature miRNAs. In addition, the TSS biogenesis process occurs so quickly that pri-miRNAs cannot be captured in sufficient numbers by RNA-Seq experiments. Because of these factors, it is difficult to identify the TSSs of miRNAs.

Earlier computational approaches to the identification of miRNA TSSs focused on sequence features, such as over-represented k-mers or CpG content, and later on the use of experimental data such as Capped Analysis of Gene Expression (CAGE) and DNase-Seq data sets [8-10]. Chien et al. were able to predict TSS locations for miRNAs using a Support Vector machine (SVM) incorporating information from CAGE tags generated by the FANTOM project [11], TSS-Seq data and H3K4me3 chromatin signatures from the CD4+ cell line [12]. PROmiRNA was able to distinguish TSS locations for miRNAs from background genomic sequences using an Expectation-Maximization (EM) algorithm incorporating information from CAGE tags generated by the FANTOM4 project and sequence features such as CpG enrichment [13]. MicroTSS identified condition-specific TSS locations for miRNAs using an SVM incorporating information from condition-specific, high-resolution RNA-Seq data and other markers such as DNase-Seq data, H3K4me3, and Pol II [14]. High-resolution RNA-Seq experimental data is not always available, and Hua et al identified condition-specific TSS locations for miRNAs using a scoring function incorporating conservation measurements and condition-specific profile of H3K4me3, DNase-Seq data that are available from ENCODE project [15]. Two problems have remained difficult to solve however: obtaining precise resolution in terms of the predicted locations of TSSs for miRNAs; and a lack of training data specifically comprised of TSS locations for miRNAs, as opposed to training data taken from the general population of TSS locations.

Deep learning models have been shown success in various bioinformatics applications such as protein structure prediction, function annotation and biomedical imaging [16, 17], but have not been explored for miRNA TSS identification. In the meantime, large-scale miRNA TSS data has recently become available from FANTOM project. In this paper, we investigate the effectiveness of using deep learning models for high-resolution miRNA TSS identification. We develop two deep learning models

including a Long-Short Term Memory Network (LSTM) model and a Convolutional Neural Network (CNN) model. Both models are trained with FANTOM benchmark data and tested against a Support Vector Machine (SVM) baseline for the purpose of identifying TSS locations for miRNAs at a resolution of 100 base pairs (bps). Our study shows that deep learning models are capable of identifying miRNA TSSs from both TSS-flanking regions and intergenic regions. Comparing with existing methods, deep learning approach shows superior performance. Methods

II. METHODS

A. Data Set Preparation

The TSS data used for model training and testing was processed from FANTOM5 project [11] as follows. TSS regions defined as CAGE tag clusters/peaks in four cell line samples: K562, GM12787, HaLa-S3 and HepG2 were first downloaded from FANTOM project. Since CAGE data have shown different peak patterns such as narrow peaks and broad peaks [18], we focused on narrow peaks for this study, i.e, tag clusters with their width smaller than 10 bps. Only narrow peaks that are at least 500bp away from others were kept, to avoid potential confusion. Because Pol II, DNase and H3K4me3 have been recognized as active markers for TSSs, we also filtered out those narrow peaks with no active markers in their 1 kb surrounding regions for each of the four cell lines. The active marker information was obtained from ENCODE project [19]. After removing redundancy caused by potential TSSs overlapping among multiple cell lines, the above procedure resulted in 7,610 narrow peaks. We further defined these narrow peaks as centers and expand them to 1kb regions. These 1 kb regions were then used as indices to extract 1 kb sequences from the HG19 human reference genome, thus forming the TSS data set for training and testing.

The positive training data set was then generated by extracting the central 100 bp regions of the above defined TSS regions. Two types of negative training data sets were separately generated for models that can predict TSSs from their flanking regions and intergenic regions respectively. Type I data set contains the pair of 100 bp regions flanking the positive training regions. Type II data set contains 100 bp regions randomly sampled from intergenic regions. Each of the two negative training data were combined with the positive training data to obtain two training data sets: the flanking region training set and the intergenic region training set. Two data sets were assembled because they present two distinct levels of difficulty. The intergenic data set was used to train models to distinguish between TSSs and regions that contain no coding or regulatory sequences, while the flanking data set was used to train models to identify TSSs from regulatory regions.

After the positive and negative samples were assembled for each data set, each sequence was converted into a 100 x 4 one-hot matrix, with each column representing the nucleotide at that position. The unknown nucleotide, N, was represented by a column of four zeros. Each sample was assigned a label of 0 for negative samples and 1 for positive samples. After assembling the two data sets, each was randomly shuffled and split into training, validation, and test sets consisting of 80%, 10%, and 10% of the samples, respectively. This process was repeated 5 times for both the flanking and intergenic data set to approximate 5-fold cross validation, because preliminary testing showed the results of both the test model and the baselines were sensitive to the random assignment of samples to each partition. This resulted in 10 data sets in total, 5 flanking data sets and 5 intergenic data sets.

B. Model Architecture

Two test models were used. The first test architecture, the LSTM model, is similar to [20] and incorporates a convolutional layer and a bidirectional LSTM (bi-LSTM) layer. The convolutional layer recognizes local motifs, while the bi-LSTM layer recognizes long-term relationships between motifs and learns a 'regulatory grammar' organizing their global spatial arrangement. The second test architecture, the CNN model, incorporates two convolutional layers to recognize local motifs, and to recognize local spatial patterns among those motifs. The test models are adapted to recognize TSSs in 100 bp sequences. The input to the test models is a 100 x 4 one-hot matrix encoding the nucleotide present at each position of the 100 bp sequence, as described in the data set preparation section, and the output is a value between 0 and 1 measuring the probability that the input sample contains a TSS. The loss function used to train the test models was binary cross-entropy and the optimizer was RMSProp [21]. The test models trained for at most 60 epochs and performed early stopping after five consecutive epochs without validation loss improvement. The test models were implemented in Python 3 using the Keras library [21]. The architecture of the test models will be described in detail in the following subsections and will be illustrated in Fig. 1.

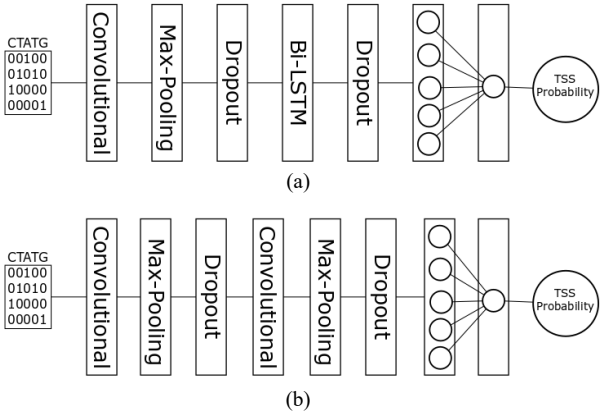


Figure 1. The architectures of the LSTM model (a) and the CNN model (b).

C. LSTM Model Architecture

The model contains eight layers in total and is detailed as follows. The first layer is a one-dimensional convolutional layer, with a window size of length 11 and 320 filters,

resulting in 320 output channels. The size of the convolution window was chosen because many of the features surrounding TSSs, such as TATA boxes, SP1-like regions, and gcg motifs, are approximately 10 bps long, allowing a single convolutional window to contain all or most of a feature. The number of output channels is the same as in the test architecture, and changes to this number did not yield improvements during preliminary testing. The activation function used by the convolutional layer was the Rectified Linear Unit (ReLU) function. The second layer is a one-dimensional max pooling layer, with both a window size and stride of length 5. This length was chosen because it maximized results during preliminary testing, and the two lengths are equal because that was the case in the test architecture and breaking that relationship was not tested during preliminary testing. The third layer is a dropout layer, used as a form of regularization during training to prevent overfitting. The dropout probability was set to 0.2. The fourth layer is a bi-LSTM layer, with 320 output channels per direction (for a total of 640 output channels). Changes to the number of output channels were found not to increase results during preliminary testing. The fifth layer is another dropout layer, also used for regularization during training and prevent overfitting. The dropout probability was set to 0.5. At this point, the tensor of output values from the bi-LSTM layer is flattened into a one-dimensional shape. The sixth layer is a dense layer with 1,000 nodes, used to provide a comprehensive assessment of the output of the bi-LSTM and produce a summary of a size more manageable for the next layer. The number of nodes was chosen because it maximized results during preliminary testing. The activation function used by the dense layer is also a ReLU function. The seventh and final layer is a dense layer containing one node, which produces the output prediction. This layer uses a sigmoid activation function, outputting a probability reflecting the model's confidence that the input sample is a positive sample.

D. CNN Model Architecture

The architecture of the CNN model is very similar to the architecture of the LSTM model, with the exception that the LSTM layer was replaced with a CNN layer and a max-pooling layer, and the probability of the second dropout layer was reduced to match the first dropout layer. The first layer is a one-dimensional convolutional layer, with a window size of length 11 and 320 filters, resulting in 320 output channels. The parameters of this layer were set the same as that of the LSTM model. The second layer is a one-dimensional max pooling layer, with both a window size and stride of length 5. The parameters of this layer were set the same as that of the LSTM model. The third layer is a dropout layer, used as a form of regularization during training to prevent overfitting. The parameters of this layer were set the same as that of the LSTM model. The fourth layer is a one-dimensional convolutional layer, with a window size of length 2 and 640 filters, resulting in 640 output channels. The size of the convolution window was chosen because this value optimized results during preliminary testing. The number of output channels was chosen to follow a convention often seen with CNNs in computer vision applications [22, 23], in which successive convolutional layers will double the number of output filters. The activation function used by the convolutional layer was the ReLU function. The fifth layer is a one-dimensional max pooling layer, with both a window size and stride of length 4. This length was chosen because it maximized results during preliminary testing. The sixth layer is another dropout layer, also used for regularization during training and prevent overfitting. The dropout probability is 0.2, which was chosen to match the value of the first dropout layer. At this point, the tensor of output values from the second convolutional and max-pooling layers are flattened into a one-dimensional shape. The seventh layer is a dense layer with 1,000 nodes, used to provide a comprehensive assessment of the output of the CNN and produce a summary of a size more manageable for the next layer. The parameters of this layer were unchanged from the LSTM model. The eighth and final layer is a dense layer containing one node, which produces the output prediction. This layer uses a sigmoid activation function, outputting a probability reflecting the model's confidence that the input sample is a positive sample, just like the LSTM model.

E. SVM Baseline

A SVM baseline introduced in Zien et al. [24] was implemented to evaluate the performance of the deep learning models for miRNA TSS prediction. The baseline employs an improved polynomial kernel function specialized to this application, which is based on the Salzberg method [25] and involves calculating the log odds of the conditional probability that nucleotide position *i* occurs within a positive sample given that nucleotide position *i-1* occurred within a positive sample versus the conditional probability that nucleotide position *i* occurs within a negative sample given that nucleotide position *i-1* occurred within a negative sample. The one-hot vectors representing the sequences in each data set are replaced with vectors containing the log odds for each position. For a pair of log odds vectors, *x* and *y*, the kernel function examines windows of length 3. It takes a weighted sum of the Euclidean distance of the log odds at corresponding positions of x and y within that window and raises that weighted sum to the power of 3. The weighted sums obtained from each window are summed to obtain one value representing the comparison of x and y.

III. Results

To determine the effectiveness with which each model could classify whether a 100 bp sequence contained a TSS or not, the test models and the SVM baseline were trained on the training set of each of the five flanking and intergenic data sets and tested on the corresponding test sets. Each of the training sets contained 18,012 training samples, and each validation and test set contained 2,252 samples each. The ratio of positive and negative samples in each partition of each data set varied slightly due to the random allocation of samples but was approximately one third positive to two thirds negative in each case. The results of each test were measured using Precision, Recall, Accuracy, and F1 Score,

and the average results across all flanking and all intergenic data sets was also collected.

A. Flanking Data Set Results

The purpose of the flanking data sets was to test each model on the more challenging task of distinguishing regions of DNA containing a TSS from neighboring regions. As is illustrated in table 1, the test models performed better than the SVM baseline on all metrics.

TABLE I. TEST RESULTS FOR FLANKING DATA SETS

Data Set	Precision	Recall	F1-Score	Accuracy
LSTM 0	0.8562	0.3671	0.5139	0.7749
LSTM 1	0.3529	0.1094	0.1670	0.6545
LSTM 2	0.9280	0.2984	0.4516	0.7358
LSTM 3	0.9437	0.2694	0.4192	0.7527
LSTM 4	0.9810	0.2640	0.4161	0.7420
Average	0.8124	0.2617	0.3959	0.7320
Average w/o LSTM 1	**0.9272**	0.2998	0.4530	**0.7513**
CNN 0	0.9302	0.3288	0.4858	0.7744
CNN 1	0.9522	0.2804	0.4332	0.7456
CNN 2	0.8725	0.3167	0.4647	0.7340
CNN 3	0.8649	0.3003	0.4458	0.7527
CNN 4	0.9809	0.2615	0.4129	0.7411
Average	0.9201	0.2975	0.4497	0.7496
Average w/o CNN 1	0.9121	**0.3018**	**0.4535**	0.7506
SVM 0	0.2283	0.2342	0.2312	0.4951
SVM 1	0.2931	0.2830	0.2879	0.5147
SVM 2	0.2969	0.2716	0.2837	0.5
SVM 3	0.2322	0.2493	0.2405	0.4782
SVM 4	0.2680	0.2423	0.2545	0.5058
Average	0.2637	0.2561	0.2598	0.4988
Average w/o SVM 1	0.2564	0.2494	0.2528	0.4948

Additionally, both test models achieved much greater precision than recall on all flanking data sets. Of the two test models, the LSTM model demonstrates greater precision while the CNN model demonstrates higher recall.

Another observation is that Flanking Data Set 1 yielded anomalously low performance for the LSTM model. As a result, the average performance of each model on every flanking data set excluding Flanking Data Set 1 was collected to illustrate the impact this had. The disparate impact of this data set on the LSTM model vs the CNN model suggests that the LSTM model may be more sensitive to how samples are assigned to each data set

B. Intergenic Data Set Results

The purpose of the intergenic data sets was to test each model on the less challenging task of distinguishing regions of DNA containing a TSS from regions of intergenic DNA. As is illustrated in table 2, the test models outperformed the SVM baseline and both of the test models achieved greater precision than recall, as was the case with the flanking datasets. It also remains the case that the LSTM model obtained greater precision than the CNN model, but the margin is smaller than it was for the flanking data sets. Unlike the flanking data sets, however, none of the intergenic data sets exhibited anomalously low performance for any of the models, so only the average of all intergenic data sets is taken here.

TABLE II. TEST RESULTS FOR INTERGENIC DATA SETS

Data Set	Precision	Recall	F1-Score	Accuracy
LSTM 0	0.972	0.3408	0.5047	0.7882
LSTM 1	0.7146	0.3981	0.5113	0.7513
LSTM 2	0.9007	0.3267	0.4795	0.7638
LSTM 3	0.9390	0.3072	0.4629	0.7620
LSTM 4	0.9289	0.2776	0.4274	0.7513
Average	**0.8911**	0.3301	0.4817	0.7633
CNN 0	0.9713	0.3324	0.4953	0.7855
CNN 1	0.9081	0.3356	0.4901	0.7718
CNN 2	0.8974	0.3267	0.4790	0.7633
CNN 3	0.8076	0.3684	0.5059	0.7598
CNN 4	0.8601	0.3347	0.4818	0.7593
Average	0.8889	**0.3395**	**0.4913**	**0.7679**
SVM 0	0.2297	0.2496	0.2392	0.4973
SVM 1	0.2154	0.2201	0.2177	0.4831
SVM 2	0.2344	0.2253	0.2298	0.4969
SVM 3	0.2304	0.234	0.2322	0.4831
SVM 4	0.2347	0.2297	0.2322	0.4920
Average	0.2289	0.2312	0.2303	0.4905

C. Results by Cell Line

The results for CNN-intergenic region training, CNN-flanking region training, LSTM-intergenic region training and LSTM-flanking region training models contain 1220, 1055, 1208 and 1025 predictions respectively. We further categorized these predictions by the four cell-lines: GM12878, HeLa-S3, HepG2 and K562 (Table 3). Note that, because most TSS regions are overlapped by cell-lines, the total number of cell-line specific predictions is larger than that of the combined predictions. Here, the evaluation on cell-relevant TSS prediction was only performed on the results obtained from deep learning models that were trained with flanking regions. This is because, unlike the prediction results obtained from models trained with flanking regions, the results from models trained with intergenic regions cannot be categorized based on the cell lines. Table 3 shows the test results by cell-lines, suggesting consistent accuracy of deep learning models across four cell lines.

We also compared this result in K562 cell to Hua et al [15]. The number of predicted TSSs in K562 cell line is 718 and 487 for our deep learning approach and Hua et al. respectively. Fig. 2 shows the sequence logos of both results,

created with WebLogo [26]. Preliminary inspection of the sequence logos reveals a TATA box pattern at the front part. Since the occurrence of TATA sequences in the close upstream regions of TSSs is the most used feature for TSS and core promoter identification [18], this pattern demonstrates that the deep learning models consistently predicts putative TSS.

Since CAGE experiments may not have captured nascent transcripts [26], we also compared the predicted TSSs with available GRO-cap data in K562 cell line from NCBI GEO database (GSM1480321). We manually checked 100 bp neighborhood of all of the false predictions and found two of the false positive predictions are indeed consistent with GRO-cap signal and might represents nascent transcriptional events. For example, [30594551, 30594601] at chromosome 6 and [141523898, 141523998] at chromosome 8 are classified as false positive, however the GRO-cap signals were observed at both locations.

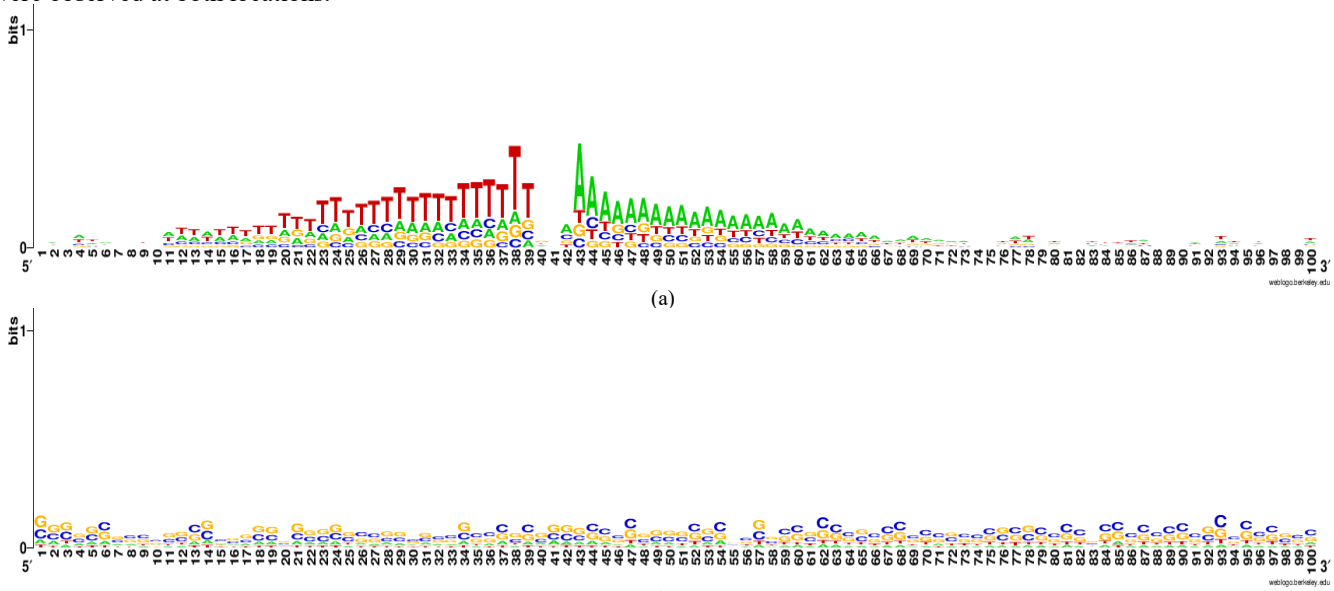
TABLE III. THE PREDICTION RESULTS BY CELL-LINES. (A) THE NUMBER OF TRUE (LEFT) AND FALSE (RIGHT) PREDICTIONS BY CELL-LINES. (B) THE ACCURACY OF PREDICTED RESULTS BY CELL-LINES

	GM12878	HeLa-S3	HepG2	K562
LSTM	623 / 57	757 / 64	637 / 65	718 / 71
CNN	616 / 57	764 / 58	628 / 59	719 / 73

(a)

	GM12878	HeLa-S3	HepG2	K562
LSTM	91.62%	92.20%	90.74%	91.00%
CNN	91.53%	92.94%	91.41%	90.78%

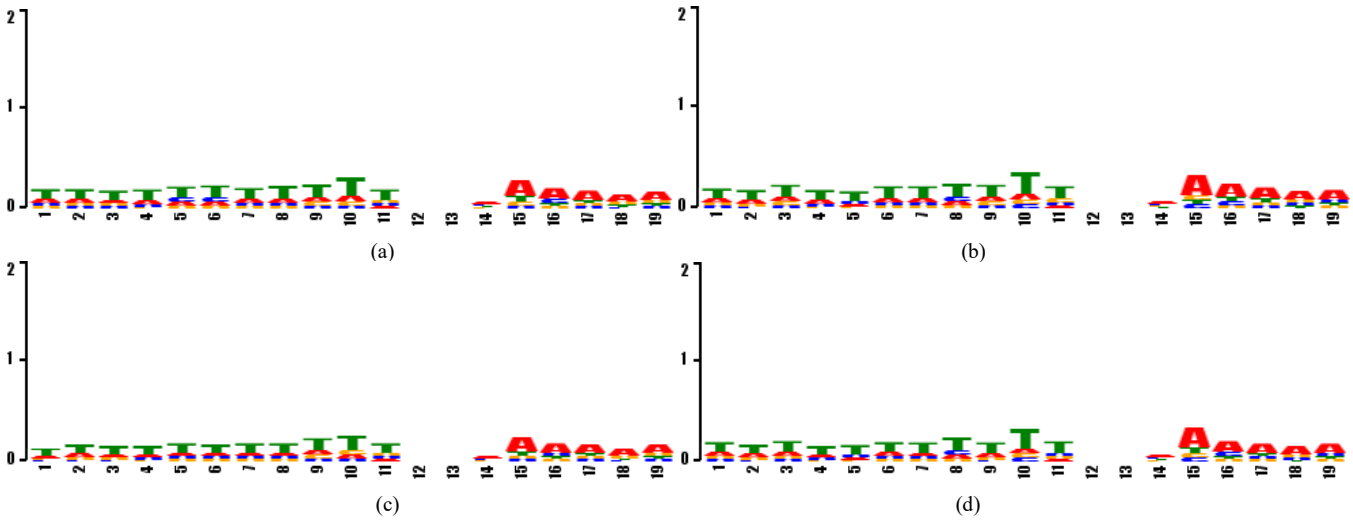
(b)



(a)

(b)

Figure 2. Sequence logos for predicted results on K562 by LSTM with flanking regions (a) and by Hua et al. (2016) (b).



(a)

(b)

(c)

(d)

Figure 3. Sequence logos for the 27 to 46 nucleotide positions of predicted positive test samples for the LSTM model on the flanking dataset (a), the LSTM model on the intergenic dataset (b), the CNN model on the flanking dataset (c), and the CNN model on the intergenic dataset (d).

IV. DISCUSSIONS

We applied deep learning models including LSTM and CNN models to identify miRNA TSSs from both their flanking regions and intergenic regions. Comparing with the base line SVM model, both LSTM and CNN models provided high-resolution miRNA TSS predictions for testing dataset obtained from FANTOM project. However, there is difference between the two deep learning models based on performance evaluation. In part, the difference between the precision and the recall achieved by the LSTM model and CNN model may be explained by the imbalanced nature of the data sets. The two to one ratio of negatives to positives may incentivize each network to learn to identify negative samples more effectively than positive samples during training. An additional factor may be the size of the data sets themselves. The trainings partition for each data set consisted of approximately 18,000 training samples, which is a small training set by the standards of typical deep learning training sets. Given the size of the data set and the imbalance of samples across classes, it may be the case that there were not enough positive examples for either network to adequately learn to identify them reliably.

In addition, the margin between precision and recall was lower for the intergenic datasets than the flanking datasets, with both the LSTM model and CNN model achieving lower precision and higher recall. One possible explanation for this is that the underlying probability distributions for positive and negative samples in the intergenic data sets is more distinct than the probability distributions for the positive and negative samples in the flanking data sets, due to the fact that the negative samples were sampled from more distinct intergenic regions rather than more similar neighboring regions. This may have had the effect of reducing the impact of the relatively small and imbalanced datasets and allowed greater recall at the expense of poorer precision.

The features recognized by the test models were examined by assembling a position weight matrix (PWM) of all samples predicted positive by the test models (illustrated in Fig. 3) and using a Pearson Chi^2 test to determine which positions showed significant deviation from background noise (represented by a uniform distribution). The results show that across both the LSTM and CNN models, and across both the flanking and intergenic datasets, the positions that showed significant departure from a uniform background were concentrated in the positions ranging from 27 to 46, the 20 positions leading up to the beginning of the TSS. The values most frequently held by these positions were T and A, indicating a TATA box. From this, it can be inferred that the test models learned to recognize a positive sample by looking for a TATA box in the 20 positions leading up to the expected position of the TSS. That the test models fixated on a single feature for identifying positive samples is interesting and may further explain the low recall scores. Any true positive sample which lacked a sufficiently clearly defined TATA box in the expected position would not have been recognized as a positive sample. This suggests a potential avenue for improving the performance of the test models: by encouraging them to consider additional features. True positive samples that lack sufficiently clear TATA boxes in the expected position may still possess other features that could be used to identify them. This encouragement could be provided by modifying the optimization function of the test models to apply a reward for recognition of diverse features or a penalty for too many predictions with the same features, or it could be provided by selecting the samples in the training set to display a broader range of features and a larger population of each.

ACKNOWLEDGMENT

This work was supported by the National Science Foundation [1356524, 1149955, 1661414]; and the National Institutes of Health [R15GM123407].

REFERENCES

[1] J. Ding, X. Li, and H. Hu, "TarPmiR: a new approach for microRNA target site prediction," *Bioinformatics,* vol. 32, pp. 2768-75, Sep 15 2016.

[2] J. Ding, X. Li, and H. Hu, "MicroRNA modules prefer to bind weak and unconventional target sites," *Bioinformatics,* vol. 31, pp. 1366-74, May 1 2015.

[3] J. Ding, X. Li, and H. Hu, "CCmiR: A computational approach for competitive and cooperative microRNA binding prediction," *Bioinformatics,* Sep 25 2017.

[4] M. Sadeghi, B. Ranjbar, M. R. Ganjalikhany, M. K. F, U. Schmitz, O. Wolkenhauer, *et al.*, "MicroRNA and Transcription Factor Gene Regulatory Network Analysis Reveals Key Regulatory Elements Associated with Prostate Cancer Progression," *PLoS One,* vol. 11, p. e0168760, 2016.

[5] Y. Wang, S. Goodison, X. Li, and H. Hu, "Prognostic cancer gene signatures share common regulatory motifs," *Sci Rep,* vol. 7, p. 4750, Jul 6 2017.

[6] Y. Lee, M. Kim, J. Han, K. H. Yeom, S. Lee, S. H. Baek, *et al.*, "MicroRNA genes are transcribed by RNA polymerase II," *EMBO J,* vol. 23, pp. 4051-60, Oct 13 2004.

[7] P. Graves and Y. Zeng, "Biogenesis of mammalian microRNAs: a global view," *Genomics Proteomics Bioinformatics,* vol. 10, pp. 239-45, Oct 2012.

[8] H. K. Saini, S. Griffiths-Jones, and A. J. Enright, "Genomic analysis of human microRNA transcripts," *Proc Natl Acad Sci U S A,* vol. 104, pp. 17719-24, Nov 6 2007.

[9] X. Zhou, J. Ruan, G. Wang, and W. Zhang, "Characterization and identification of microRNA core promoters in four model species," *PLoS Comput Biol,* vol. 3, p. e37, Mar 9 2007.

[10] A. Marson, S. S. Levine, M. F. Cole, G. M. Frampton, T. Brambrink, S. Johnstone, *et al.*, "Connecting microRNA genes to the core transcriptional regulatory circuitry of embryonic stem cells," *Cell,* vol. 134, pp. 521-33, Aug 8 2008.

[11] M. Lizio, J. Harshbarger, H. Shimoji, J. Severin, T. Kasukawa, S. Sahin, *et al.*, "Gateways to the FANTOM5 promoter level mammalian expression atlas," *Genome Biol,* vol. 16, p. 22, Jan 5 2015.

[12] C. H. Chien, Y. M. Sun, W. C. Chang, P. Y. Chiang-Hsieh, T. Y. Lee, W. C. Tsai, *et al.*, "Identifying transcriptional start sites of human microRNAs based on high-throughput sequencing data," *Nucleic Acids Res,* vol. 39, pp. 9345-56, Nov 2011.

[13] A. Marsico, M. R. Huska, J. Lasserre, H. Hu, D. Vucicevic, A. Musahl*, et al.*, "PROmiRNA: a new miRNA promoter recognition method uncovers the complex regulation of intronic miRNAs," *Genome Biol,* vol. 14, p. R84, Aug 16 2013.

[14] G. Georgakilas, I. S. Vlachos, M. D. Paraskevopoulou, P. Yang, Y. Zhang, A. N. Economides*, et al.*, "microTSS: accurate microRNA transcription start site identification reveals a significant number of divergent pri-miRNAs," *Nat Commun,* vol. 5, p. 5700, Dec 10 2014.

[15] X. Hua, L. Chen, J. Wang, J. Li, and E. Wingender, "Identifying cell-specific microRNA transcriptional start sites," *Bioinformatics,* vol. 32, pp. 2403-10, Aug 15 2016.

[16] K. Lan, D. T. Wang, S. Fong, L. S. Liu, K. K. L. Wong, and N. Dey, "A Survey of Data Mining and Deep Learning in Bioinformatics," *J Med Syst,* vol. 42, p. 139, Jun 28 2018.

[17] S. Min, B. Lee, and S. Yoon, "Deep learning in bioinformatics," *Brief Bioinform,* vol. 18, pp. 851-869, Sep 1 2017.

[18] P. Carninci, A. Sandelin, B. Lenhard, S. Katayama, K. Shimokawa, J. Ponjavic*, et al.*, "Genome-wide analysis of mammalian promoter architecture and evolution," *Nat Genet,* vol. 38, pp. 626-35, Jun 2006.

[19] "The ENCODE (ENCyclopedia Of DNA Elements) Project," *Science,* vol. 306, pp. 636-40, Oct 22 2004.

[20] D. Quang and X. Xie, "DanQ: a hybrid convolutional and recurrent deep neural network for quantifying the function of DNA sequences," *Nucleic Acids Res,* vol. 44, p. e107, Jun 20 2016.

[21] F. a. o. Chollet. (2015). *Keras*. Available: https://keras.io

[22] K. Z. Simonyan, A., "Very deep convolutional networks for large-scale image recognition.," in *International Conference on Learning Representations*, 2015.

[23] K. He, X. Zhang, S. Ren, and J. Sun., "Deep residual learning for image recognition.," in *CVPR*, 2016, pp. 770–778.

[24] A. Zien, G. Ratsch, S. Mika, B. Scholkopf, T. Lengauer, and K. R. Muller, "Engineering support vector machine kernels that recognize translation initiation sites," *Bioinformatics,* vol. 16, pp. 799-807, Sep 2000.

[25] S. L. Salzberg, "A method for identifying splice sites and translational start sites in eukaryotic mRNA," *Comput Appl Biosci,* vol. 13, pp. 365-76, Aug 1997.

[26] Q. Liu, J. Wang, Y. Zhao, C. I. Li, K. R. Stengel, P. Acharya*, et al.*, "Identification of active miRNA promoters from nuclear run-on RNA sequencing," *Nucleic Acids Res,* vol. 45, p. e121, Jul 27 2017.