02-710 Final Report
Optimize ML Models to find transcription factor binding site

Xinling Li and Zhenyu Yang

1 Objective

The goal of this project is optimize machine learning methods to find transcription factor binding sites. In our project, we implemented CNN, RNN, and HMM and apply them to the same data set to see their performance.

2 Introduction

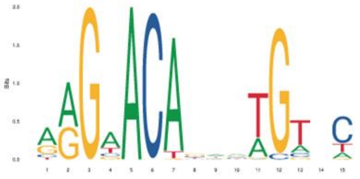
Predicting transcription factor binding sites is very important for gene regulatory network. A lot of tools that have such function such as GLAM2 doesn't assume nucleotide dependency. Also, tools such as JASPAR can't have more than one sequence as input at a time. However, transcription factor binds to enhancer which can be 1Mbp far away from the gene[1][2]. Therefore, HMM and deep neural network such as RNN will be very useful in predicting transcription factor binding sites. RNN has proven to be very successful in natural language processing and DNA sequences are similar to text in that each character has long-term dependency. Also, it was found that CNN can extract layers of translational-invariant feature maps. Therefore, in this project, we mainly focus on using these three methods to identify transcription factor binding sites.

3 Data resource:

All data are downloaded from JASPER: http://jaspar.genereg.net.

Training data and test data: TFBS of human steroid hormone receptor NR3 (positive) and non-TFBS of NR3 (negative).

The TFBS sequence logo of NR3 is shown on the right. [6]



Data sets can be described as the following table:

	Perfect data	Imperfect data
Training set	9,500 positive, 9,500 negative(no overlap*)	9,500 positive, 19,000 negative(partially overlap*)
Validation set	500 positive, 500 negative(no overlap*)	500 positive, 1000 negative(partially overlap*)
Test set	1,028 positive, 1,028 negative(no overlap*)	1,028 positive, 2,529 negative(partially overlap*)

Table 1

All the sequences are 15-nucleotide long.
*No overlap: DNA sequence that doesn't contain substrings that are the same as the ones in known positive sequence
*Partially overlap: DNA sequence that contain substrings that are the same as the ones in known positive sequence and the length of overlapping region(substring) can be between 1-14.

4 Methods

Part I Feature extraction:

One hot encoding:

Convert short DNA sequences of length n (categorical features) into a sequence of binary numbers of length 4n. For example, "ATCG" is converted to 1000010000100001.
Every four digits represent a single character in the DNA sequence.

Part II Machine learning methods:

1) Convolutional neural network(CNN).

The hidden layers of a CNN typically consist of convolutional layers, pooling layers, fully connected layers and normalization layers. Our neural network is based on 1D convolutional layers (Fig. 1):

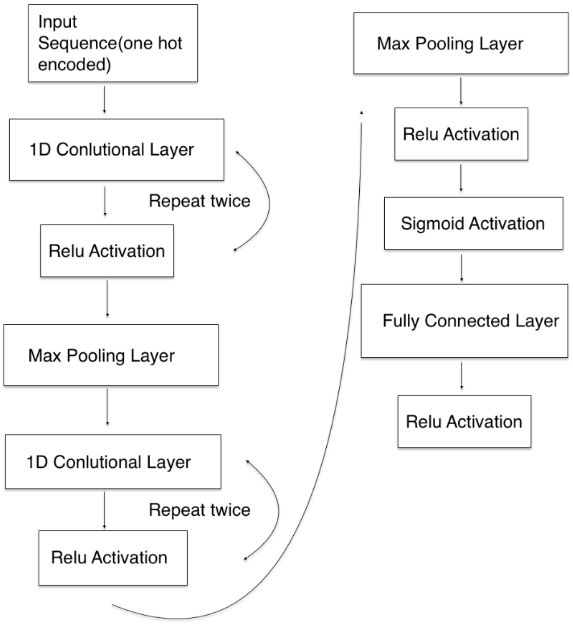


Figure 1

2)Recurrent Neural Network.

In RNN the hidden layers are recurrent layers, where every neuron is connected to every other neuron in the layer.

LSTM(long short term memory) is a very important step in RNN. An LSTM cell stores a value (state) for either long or short time periods.It consists forget gate, input gate and output gate(Fig2)[3].

Fully connected layer:

- Forget gate: $\boldsymbol{f}^{(t)} = \sigma(\boldsymbol{W}_{fx}\boldsymbol{x}^{(t)} + \boldsymbol{W}_{fh}\boldsymbol{h}^{(t-1)} + \boldsymbol{b}_f)$
- Input gate: $\boldsymbol{i}^{(t)} = \sigma(\boldsymbol{W}_{ix}\boldsymbol{x}^{(t)} + \boldsymbol{W}_{ih}\boldsymbol{h}^{(t-1)} + \boldsymbol{b}_i)$
- Output gate: $\boldsymbol{o}^{(t)} = \sigma(\boldsymbol{W}_{ox}\boldsymbol{x}^{(t)} + \boldsymbol{W}_{oh}\boldsymbol{h}^{(t-1)} + \boldsymbol{b}_o)$
- New cell state: $\boldsymbol{c}^{(t)} = \boldsymbol{f}^{(t)} \,\mathrm{o}\, \boldsymbol{c}^{(t-1)} + \boldsymbol{i}^{(t)} \,\mathrm{o}\, \boldsymbol{c}^{(t-1)} \tanh(\boldsymbol{W}_{cx}\boldsymbol{x}^{(t)} + \boldsymbol{W}_{ch}\boldsymbol{h}^{(t-1)} + \boldsymbol{b}_c)$
- Unit's output: $\boldsymbol{h}^{(t)} = \boldsymbol{o}^{(t)} \,\mathrm{o}\tanh(\boldsymbol{c}^{(t)})$

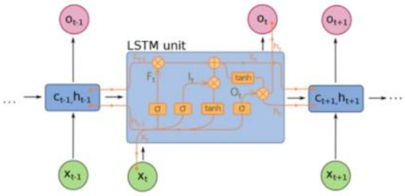


Figure 2

We design our RNN model as the figure below:

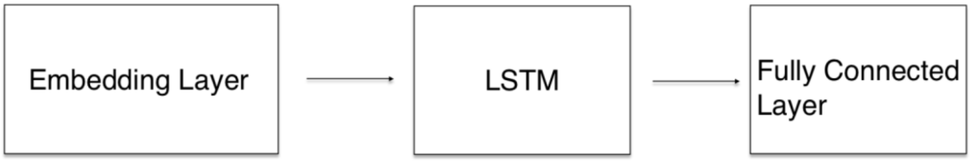


Figure 3

3) Hidden Markov Model:

Hidden Markov Model (HMM) is a statistical Markov model in which the system being modeled is assumed to be a Markov process with unobserved (i.e. hidden) states. There are nth order HMM(Figure 4)[4], which means one state emission probability is dependent on the n states after the current state. In this assignment, we choose 2nd order HMM model.

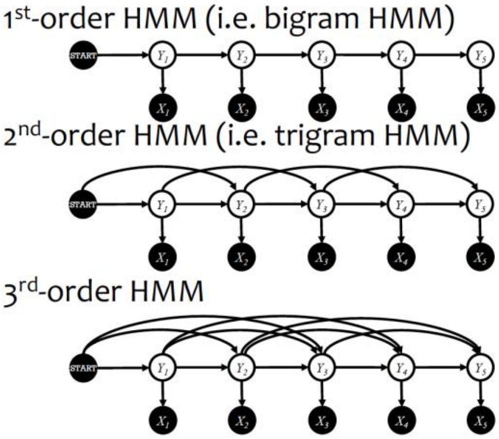


Fig 4

5.Results

1)Perfect data

The training data set consists of 19,000(9,500 positive, 9,500 negative) samples and the test data set contains 2056(1028 positive, 1028 negative) samples that extracted from the raw data set. The length of each sequence is 15 nucleotides.

The positive samples should be the ones that are sure to be transcription factor binding motif. The negative samples are the ones that are sure not to be transcription factor binding motif randomly chosen in the downstream or upstream of a sequence, which are don't have overlap with the positive ones.

i. CNN model :

Plot with epochs vs accuracy and epochs vs loss. For the training data set, accuracy after 20 epochs is 0.9916 and for validation set accuracy after 20 epochs is 0.9890. The Loss is for training set and validation set is 0.0282,0.0447(Fig.5). The test accuracy is 0.9859.

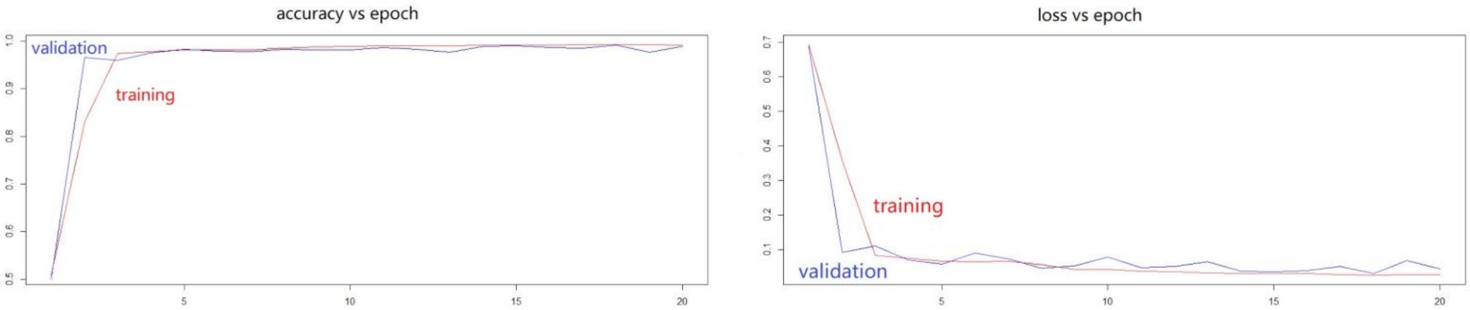
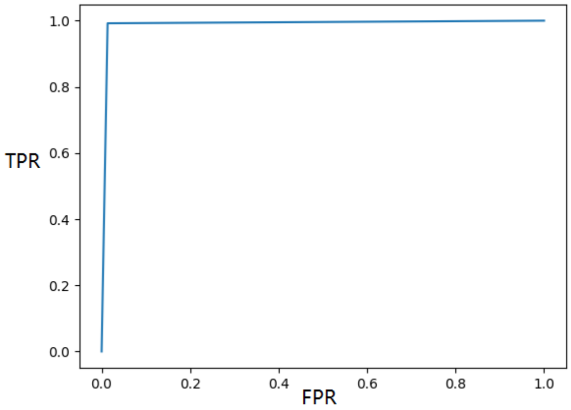


Figure 5



ROC curve

TPR: true positive rate

FPR: false positive rate

ii. RNN model:

Training accuracy is 0.9953 and validation accuracy is 0.998 after 10 epochs. Loss for train and test are 0.0244 and 0.0091(Figure 6). The test accuracy is 0.9985.

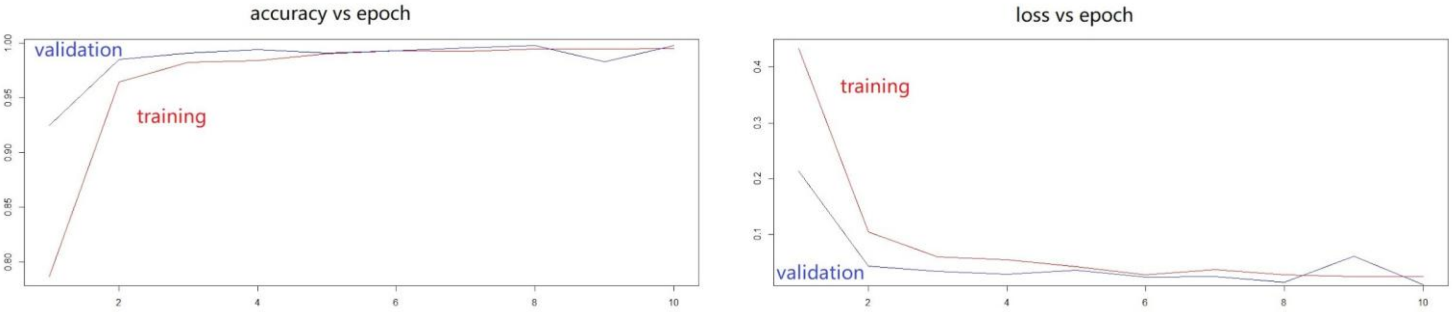
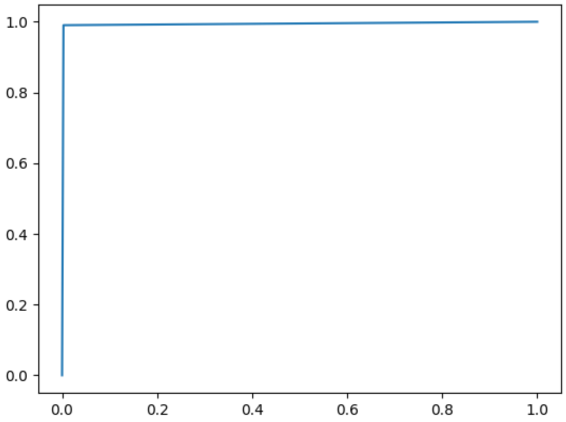


Figure 6



ROC curve(TPR vs FPR)

iii. HMM:

We did 100 iterations and the train accuracy is 0.995 and test accuracy is 0.991.

2) Imperfect data

In reality, the sequences cannot be this perfect. Usually when you generated a sequence, you hardly know whether a negative sequence contains part of a positive sequence. That is to say, a negative sequence can be partially positive.

The new data set (each strand length is 15 nucleotides) is designed as follows. Training data set consists of 28,500 (9,500 positive, 19,000 negative) samples and test data contains 3557 (1028 positive, 2529 negative) samples. The negative samples can have maximum 14 nucleotides overlap with the positive strand. In addition, we add some positive samples which are known transcription factor binding sites (1000 to the training set, 500 to the test set) from another transcription factor in human to be the negative samples in the current data set.

i) CNN model.

Training accuracy is 98.69% and validation accuracy is 95.75% after 20 epochs(Fig.7). Loss for train and test are 0.0574 and 0.1443. The test accuracy is 97.73%

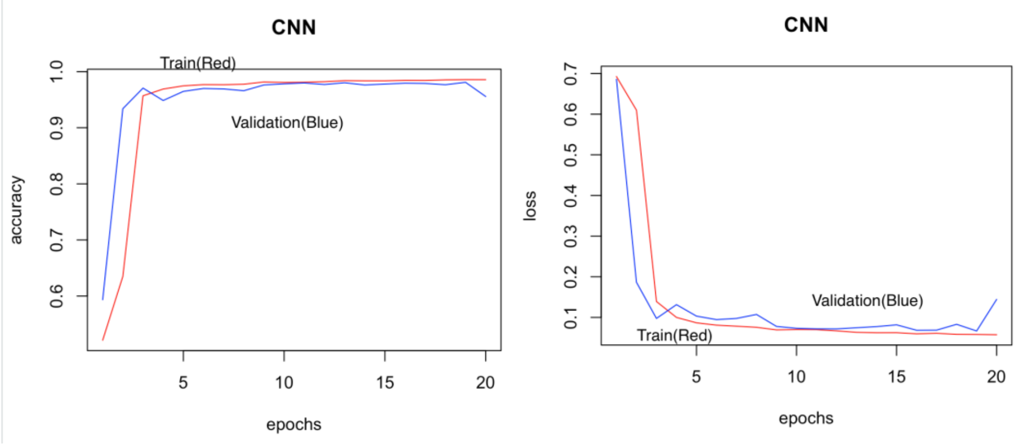
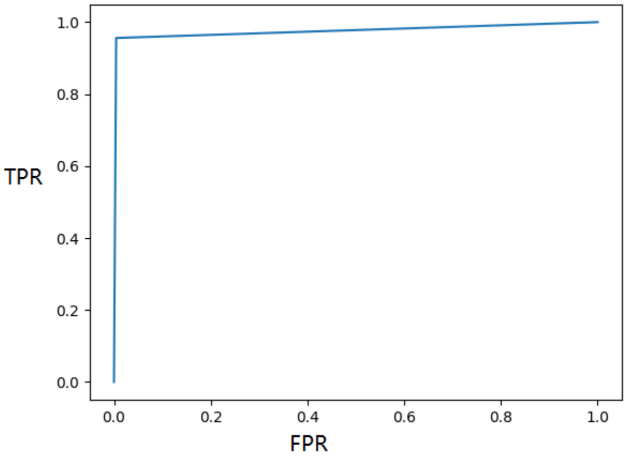


Figure 7



ii) RNN model

Training accuracy is 0.9731 and validation accuracy is 0.988 after 10 epochs(Fig.8). Loss for train and test are 0.1266 and 0.0631. The test accuracy is 0.9883.

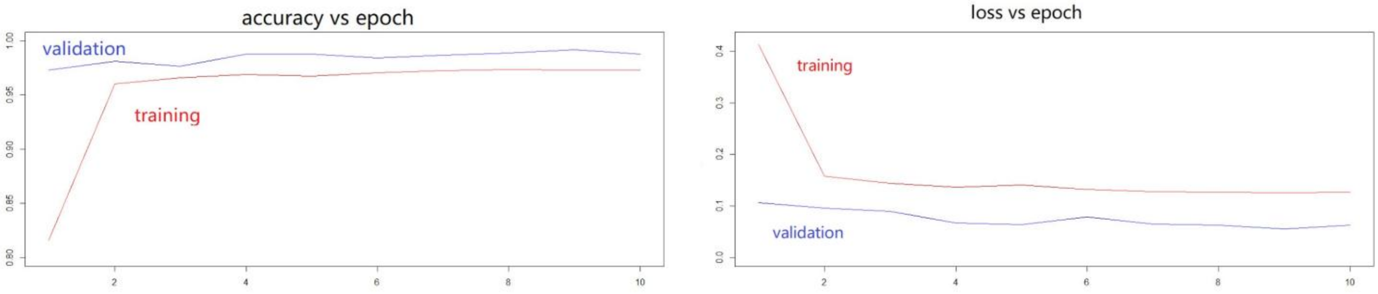
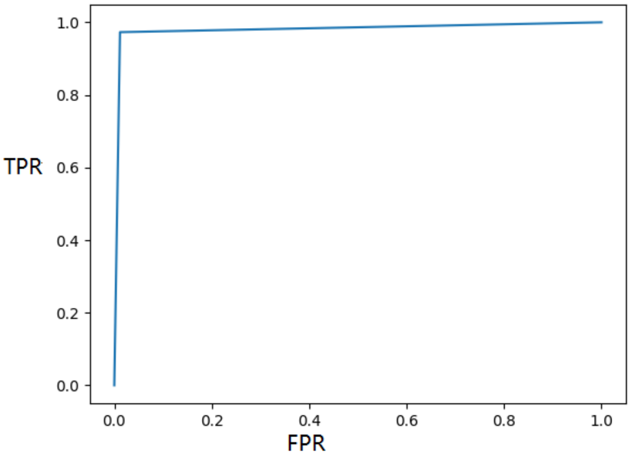


Figure 8



ROC curve

iii) HMM model

After 100 iterations, the train accuracy is 82.76% ,test accuracy is 82.04%.

6. Discussion:

In our project, training set, validation set, and test set are relative large, which is ideal for artificial neural network. We validate our model to prevent overfit. When the validation accuracy no longer increases, we stop training our model. Then we use the test set to test the model.

In our perfect training set, there is 1:1 ratio of positive and negative examples. In our imperfect training set, the number of negatives is twice the amount of positives. We design our data set in this way because in the real world, there are only small amount of sequences that are transcription factor binding site. We noticed that After adding some ‘noise’ to the data, the training accuracy, validation accuracy, and test accuracy all go down for 3 models. The noise had most impact on HMM, for which the test accuracy is only 82.04%. In reality, especially for the new generated sequence, there should be more noise.

The overall performance of RNN is the best among all three methods. The performance of second-order HMM is better than CNN under ideal situation. The result makes sense since RNN take long-term dependency into account, while second-order HMM only considers two adjacent nucleotides. CNN performs the worst under ideal condition, because it's a feed-forward neural network, which is best at image processing.

However, one drawback of RNN is that it takes relatively long time to run each epoch. HMM has the shortest running time, but the accuracy of HMM in predicting TFBS in imperfect data set is relatively low.

In a word, we think that CNN and HMM are not suitable for sequence analysis, especially for finding TFBS, since CNN does not considering nucleotides' dependency and HMM only consider short term dependencies.

7. Future development:

In our project, we only take sequence of nucleotide into consideration. However, in reality, transcription factor is cell specific. For example, a transcription factor might activate only a set of genes needed in certain neurons [5]. If time allows, we will collect more gene data from different types of cell to validate our approach.

8. Conclusion:

RNN model is the best in predicting transcription factor binding site.

9. Citation:

[1] Blackwood, E. M.; Kadonaga, J. T. (1998). "Going the Distance: A Current View of Enhancer Action". *Science*. 281 (5373): 60–3. doi:10.1126/science.281.5373.60. PMID 9679020.

[2]Pennacchio, L. A.; Bickmore, W.; Dean, A.; Nobrega, M. A.; Bejerano, G. (2013). "Enhancers: Five essential questions". *Nature Reviews Genetics*. 14(4): 288–95. doi:10.1038/nrg3458. PMC 4445073 . PMID 23503198.

[3] Slides from Roi Yehoshua, http://www.cs.biu.ac.il/~yehoshr1/

[4] Slides from professor Matt Gormley

[5] https://www.khanacademy.org/science/biology/gene-regulation/gene-regulation-in-eukaryotes/a/eukaryotic-transcription-factors

[6] JASPAR http://jaspar.genereg.net/matrix/MA0007.2/