Fully Interpretable Deep Learning Model of Transcriptional Control

Yi Liu

University of Chicago

Special Thanks: John Reinitz and Kenneth Barr

Outline

Introduction

The problem:
How do you get from **sequences** *to* **mRNA** *expression in multicelluar organism?*

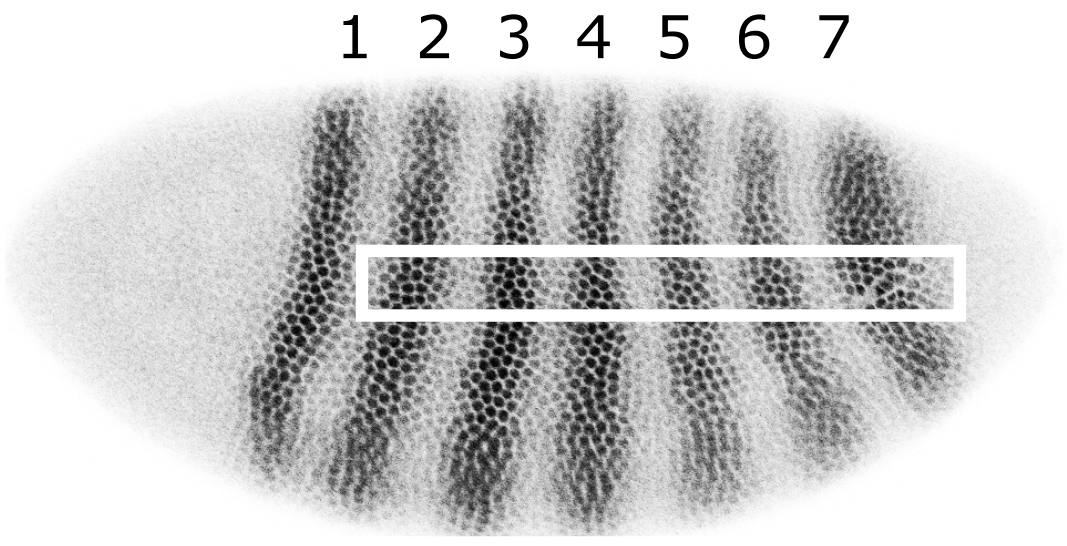

Figure 1: A Drosophila embryo about 3 hours after fertilization which has been stained for Eve protein

Main Contribution

We demonstrate that this process can be described by a Deep Neural Networks.

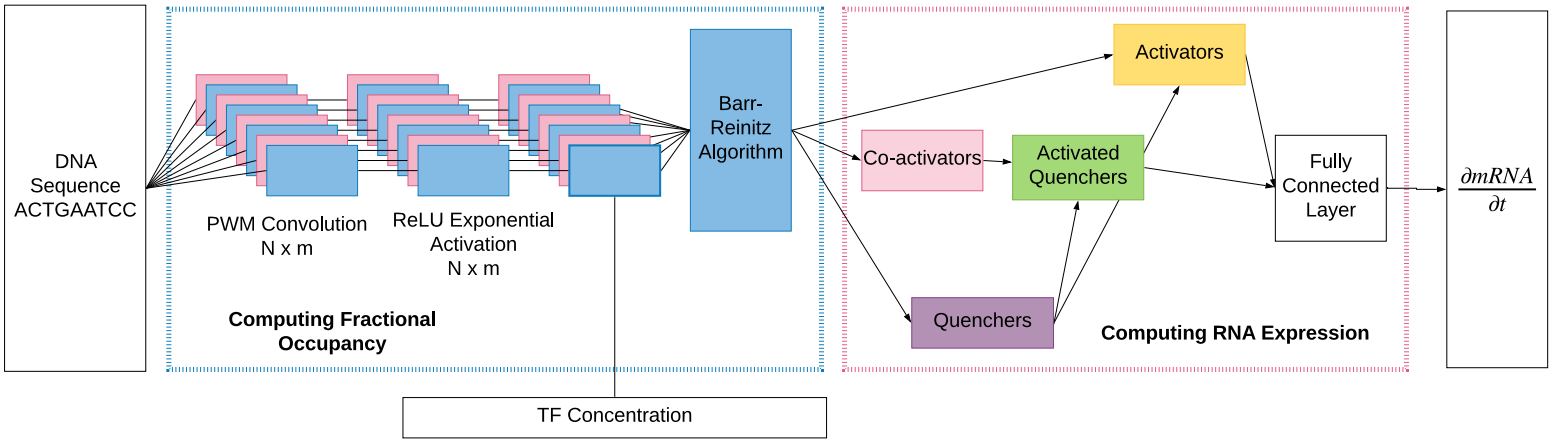

Figure 2: A graphical representation of the DNN.

Basic Structure

A Deep Neural Network is made up of linear activation units.

$$y_{i,j} = \underbrace{\sigma}_{\text{non-linear activation}} \left(\underbrace{\sum_k W_{k,j} X_{k,i} + \texttt{bias}}_{\text{linear part}} \right)$$

Here the non-linear activation can be any non-linear functions.
The key is to construct a layered structure that keep feeding forward.

Convolutional Neural Networks

Convolutional Neural Network are a special kind of linear activation units with a $l \times k$ filter W on an $n \times m$ X matrix. The filter moves across the matrix with, doting with entries of the matrix and outputting values behind. For example: $W = \begin{pmatrix} 1 & 2 \\ 2 & 1 \end{pmatrix}$ and $X = \begin{pmatrix} 1 & 1 & 1 & 2 \\ 3 & 1 & 1 & 3 \end{pmatrix}$ The resultant output is matrix

$$\texttt{Conv}(X, W) = (\underbrace{10}_{1+6+2+1} \quad \underbrace{6}_{1+2+2+1} \quad \underbrace{10}_{1+2+4+3})$$

Recurrent Neural Network

Recurrent Neural Networks are another special kind of neural network where there is space or time component to the object that you want to model. You combine the results from the previous time with the new input from the current time to produce a prediction and a output for the next time stamp.
Hidden Markov model can be seen as a recurrent neural network in some cases.

DNN Interpretation of the Transcription Model

Find Binding Sites using PWMs

An indicator representation of DNA sequence is used as input. For example, if we have a sequence of ACTNGTTA, the corresponding matrix is

$$\begin{pmatrix} \text{A} & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ \text{C} & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ \text{G} & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ \text{T} & 0 & 0 & 1 & 0 & 0 & 1 & 1 & 0 \\ \text{N} & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \end{pmatrix}$$

PWM here is essentially a convolutional kernel to the matrix, the output are scores S_i of the (potential) binding sites that starts at indices i in the sequence.

Affinity

From the Score we need to find the affinity at each binding. To compute this we use basic chemistry. But the resultant chemistry can be easily identified as having a neural network structure.

$$K_{i,i+m;a} = \underbrace{\exp\left(-\frac{(\overbrace{\texttt{ReLU}(S_{i,i+m;a})}^{\texttt{X}} - \overbrace{S_{\max;a}}^{\texttt{bias}})}{\underbrace{\lambda_a}_{\texttt{Weight}}}\right)\mathbb{I}(S_{i,i+m;a} > 0)}_{\texttt{activation}}, \tag{1}$$

Here a is the TF name, i is the sequence index and m is the size of the protein as specified by the PWM

Fractional Occupancy

Combining the $K_{i,i+m;a}$ with the concentration of the TF at particular site of Drosophilia gives us the free energy of the TFs and each binding site.

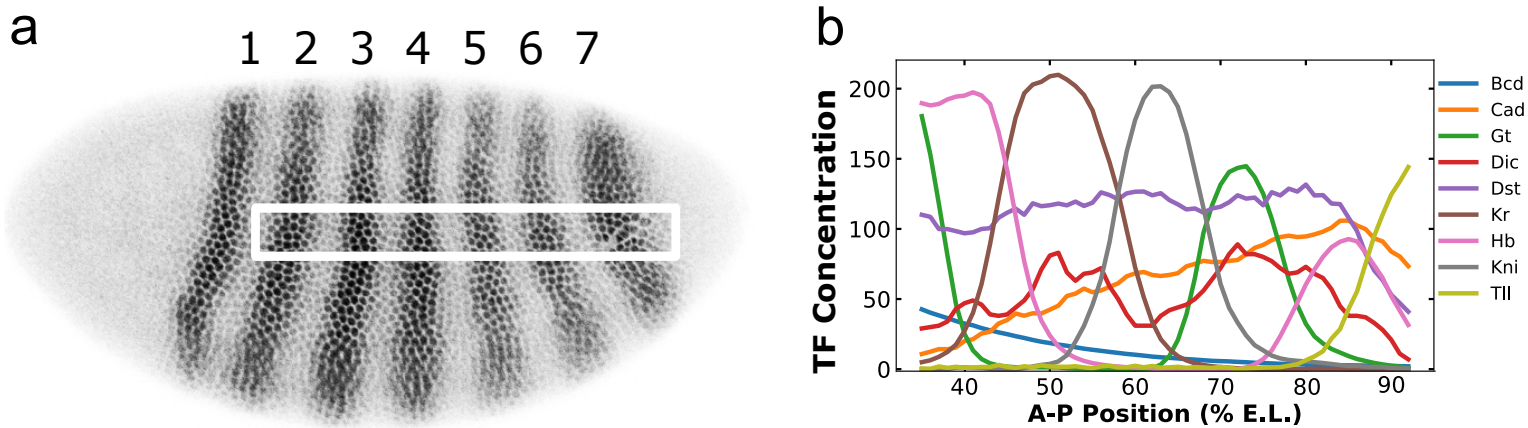

Figure 3: Concentration of the TFs across the organism

However, to know exactly how much TFs are at a particular binding site, we need to consider all possible combination of the TF arrangements and also the fact that some TFs are more likely to be together (Cooperativity).

Barr-Reinitz Algoritm

Algorithm 1 The Algorithm for Fractional Occupancy

Initialize $Z_N^- = 1$ and $Z_0^+ = 1$
for $i \leftarrow 1$ to N **do**
  $q_{i,a} = [TF]_a K_{i,a}$
  for $a \in \{\text{Transcription Factors}\}$ **do**
    $Z_{i,a}^{+nc} = q_{i,a} Z_{i-m-1}^+$
    $Z_{N-i,a}^{-nc} = q_{N-i,a} Z_{N-i+m+1}^-$
    $Z_{i,a}^{+c} = \sum_{j=m+1}^{c_d} w_{ij}^{\text{coop}} q_{i,a} q_{i-j,a} Z_{i-j-m-1}^+$
    $Z_{i,a}^{-c} = \sum_{j=m+1}^{c_d} w_{ij}^{\text{coop}} q_{N-i,a} q_{N-i+j,a} Z_{N-i+j+m+1}^-$
  end for
  $Z_i^+ = \sum_a Z_{i,a}^{+nc} + Z_{i,a}^{+c}$
  $Z_{N-i}^- = \sum_a Z_{i,a}^{-nc} + Z_{i,a}^{-c}$
end for
return $Z_a^{+nc}, Z_a^{+c}, Z_a^{-nc}, Z_a^{-c}, Z_0^-$

Essentially

This is essentially a dynamic programming algorithm but this can be easily turned into a forward and backward recurrent neural network because what you need to know about the fractional occupancy of the current site is just dependent on previous sites.
You can see the sequence as a 1-dimensional space domain where you move across to output different fractional occupancy at each point.

Exploring TF-TF interactions

We have three kind of interactions for this model

1. Co-activation
2. Quenching
3. Activation
   (In more general cases there are)
4. Co-Quenching
5. Long-range activation
6. Long-range quenching

Exploring TF-TF interactions

Co-activation:

$$\hat{f}_i^{QC} = f_i^{QC} \prod_{j=i-k}^{i+k} (1 - d_c(j) E_C^{QC} f_j^C), \tag{2}$$

$$\hat{f}_i^{QC} \approx f_i^{QC} \prod_{j=i-k}^{i+k} (1 - E_C^{QC} f_j^C)^{d_c(j)}$$

$$\Rightarrow \hat{f}_i^{QC} = \exp\left(\sum_{j=i-k}^{i+k} d_c(j) \log(1 - E_C^{QC} f_j^C)\right) f_i^{QC} \tag{3}$$

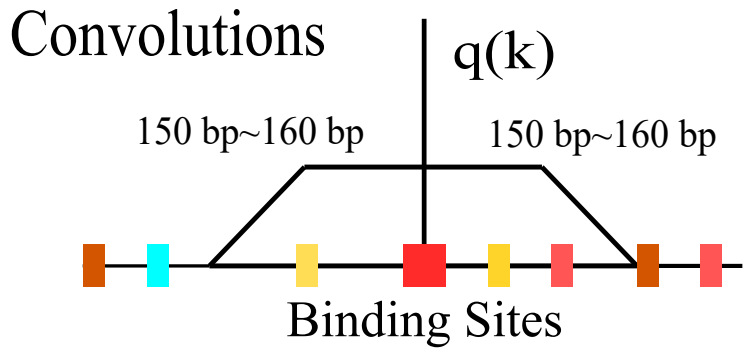

Figure 4: Coactivation can be seen as convolutions

Exploring TF-TF interactions

Quenching:

$$\hat{f}_i^A = f_i^A \prod_{j=i-k}^{i+k} (1 - d_q(j) E_A^Q f_j^Q), \tag{4}$$

$$\hat{f}_i^A \approx \exp\left(\sum_{j=i-k}^{i+k} d_q(j) \log(1 - E_A^Q f_j^Q) \right) f_i^Q. \tag{5}$$

This gives three convolutional linear activation units.

$$y_j = \log(1 - E_A^Q f_j^Q); \quad z_i = \exp\left(\sum_{j=i-k}^{i+k} d_q(j) y_j \right); \quad \hat{f}_i^Q = f_i^A z_i. \tag{6}$$

Exploring TF-TF interactions

Activation:

$$[\text{mRNA}] \propto R \left(\frac{\exp\left(\sum_{j \in \{A, Q_C\}} E_j \overbrace{\sum_i \hat{f}_i^j}^{\texttt{X}} - \overbrace{\theta}^{\texttt{bias}} \right)}{1 + \exp\left(\sum_{j \in \{A, Q_C\}} E_j \sum_i \hat{f}_i^j - \theta \right)} \right). \quad (7)$$

This can be seen as fully connected layer with a Sigmoid Activation.

Training Process

You can code this into Tensorflow or Keras and you can traing this using Stochastic Gradient Descent. We use a special form of SGD called ADAM with Nestrov Acceleration.

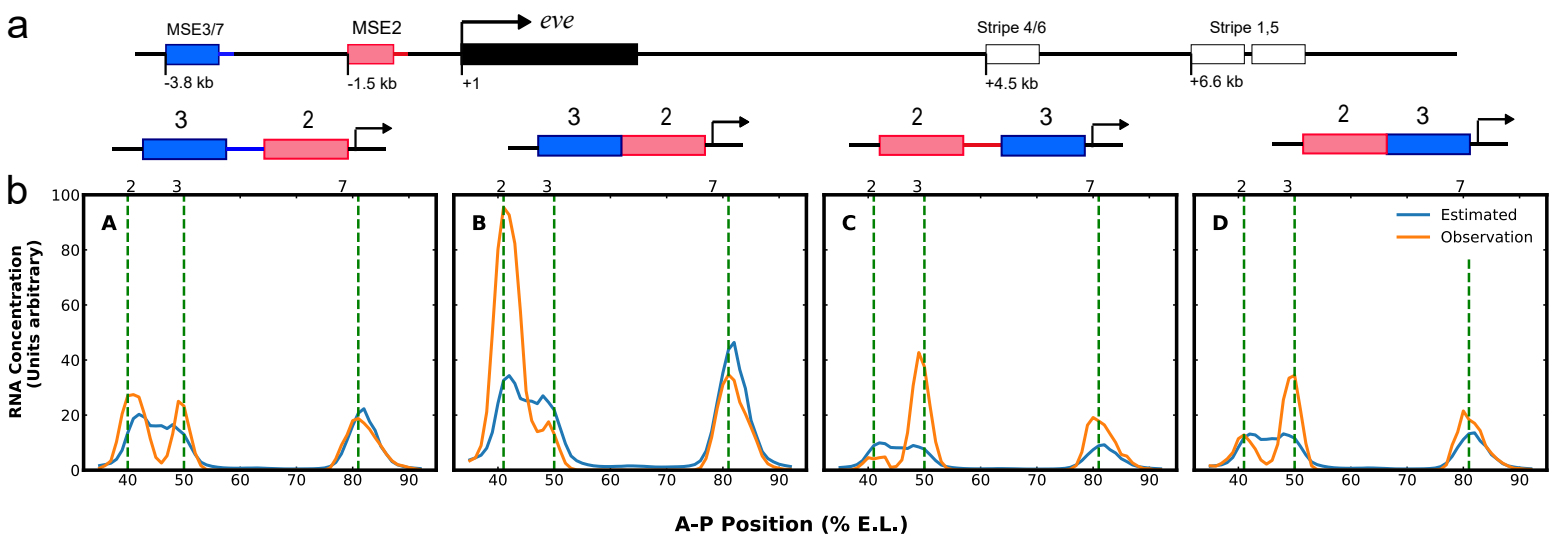

Figure 5: Training

Prediction Results

We predict our methods using sequences from other Fries and other enhancers and this is our results.

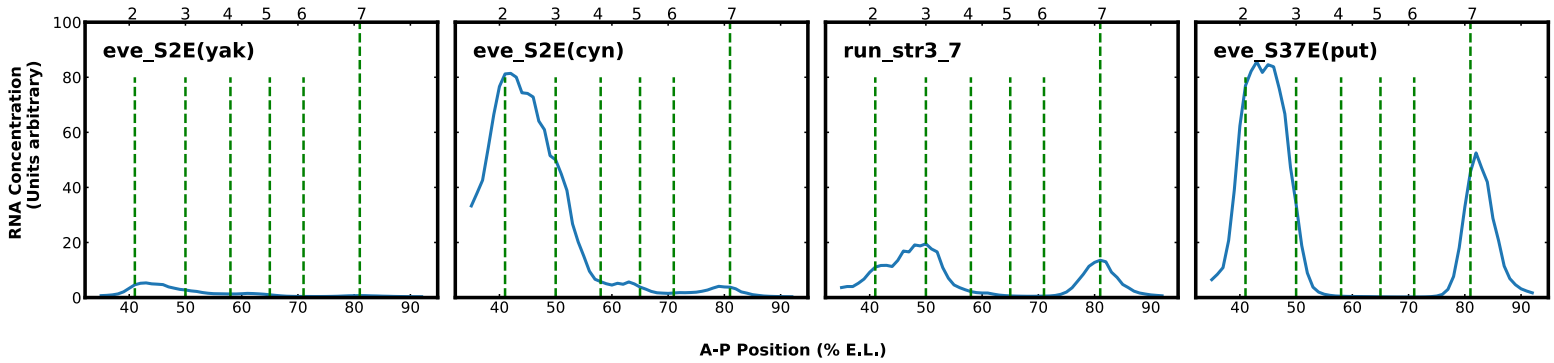

Figure 6: The figure shows four examples of predictions driven by enhancers not used for training. The location of eve stripes 2 through 7 are shown by vertical dashed lines. The vertical axis shows predicted mRNA concentration and horizontal axis shows A-P position in % E.L. The enhancers shown are described in the text.

These predictions are generally accurate.

Future Work

This work has shown the following:

1. DNN to model Transcriptions is not bad idea
2. We have the first non-trivial DNN that is full interpretable
3. We will have the potential of using new methods of optimization (such as reinforce or Q-Learning) genomic scale problem.

Next Step:

1. Use on a data of a genomic scale.
2. Further improve on the flexibility of the code and model to encompass more ideas.

Special Thanks

C o-authors:
Kenneth Barr
Department of Human Genetics
University of Chicago
Chicago IL USA
barr@uchicago.edu
John Reinitz
Departments of Statistics,
Ecology and Evolution,
Molecular Genetics & Cell Biology
Institute of Genomics and Systems Biology
University of Chicago
Chicago IL USA
reinitz@uchicago.edu
This work was supported by grant R01 OD10936 from the US NIH

References

i Liu, Kenneth Barr, John Reinitz (2019)

Fully Interpretable Deep Learning Model of Transcriptional Control

Submitted (doi: https://doi.org/10.1101/655639)