

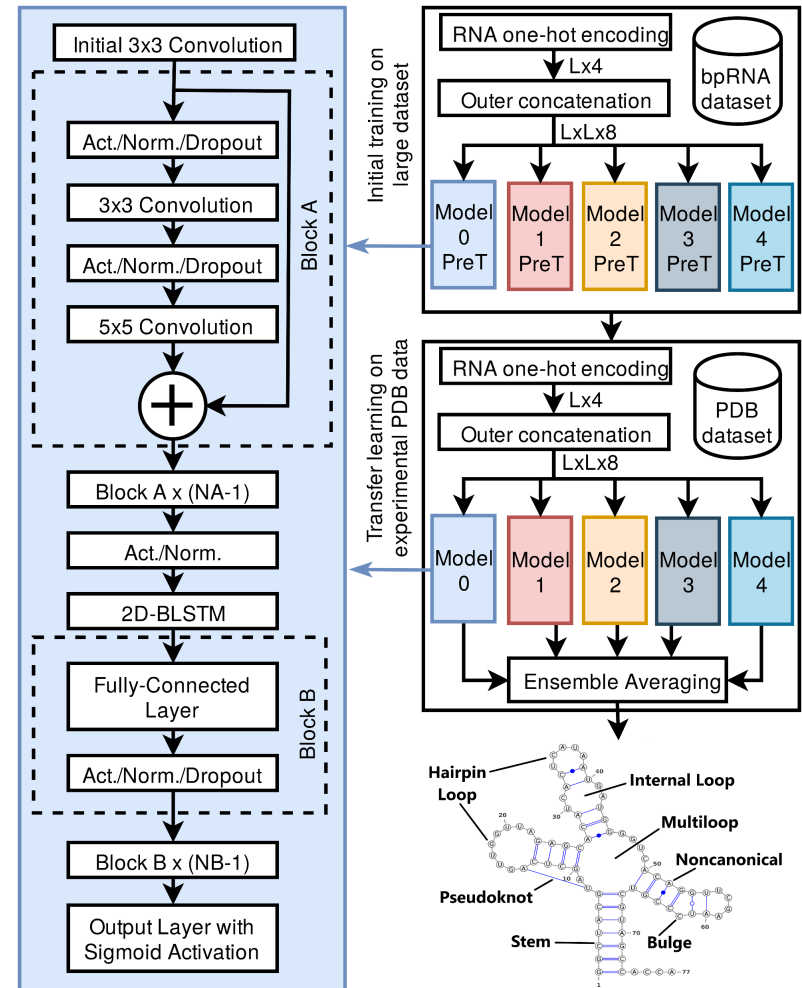
SPOT-RNA: RNA Secondary Structure Prediction using an Ensemble of Two-dimensional Deep Neural Networks and Transfer Learning

- Ensemble of 5 models (more and more convolutions from model 0-4)
- BiLSTM in Model 3
- Currently only code for prediction is public
- The goal of the model is to predict the pairing probability of each base pairs
- Use the RNA structures from **bpRNA database** for **initial training**, then use the structures from the **PDB database** for **transfer training**.

	training	validation	testing
bpRNA	10815	1301	1306
PDB	121	68+40	31

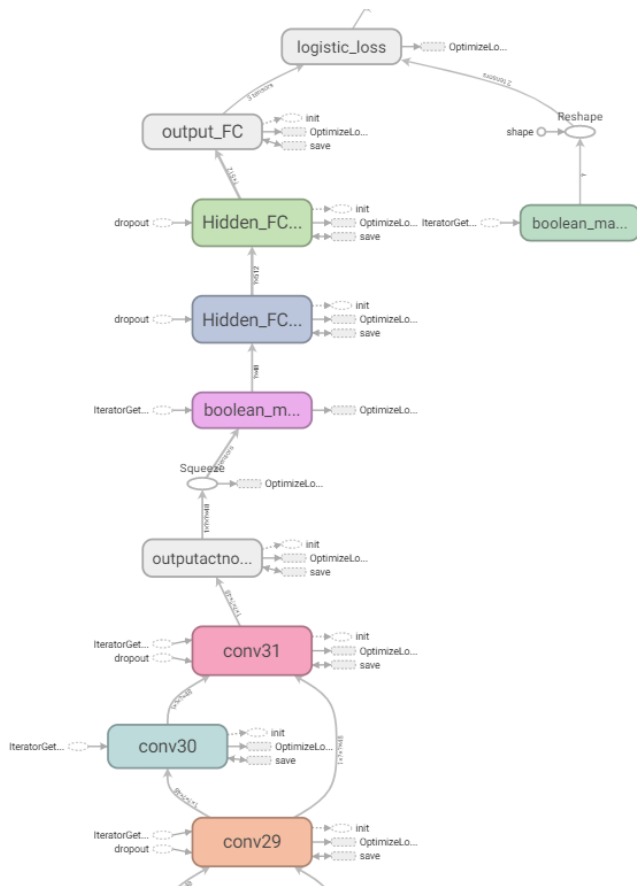
bpRNA median: ~100nt

PDB median: ~70nt

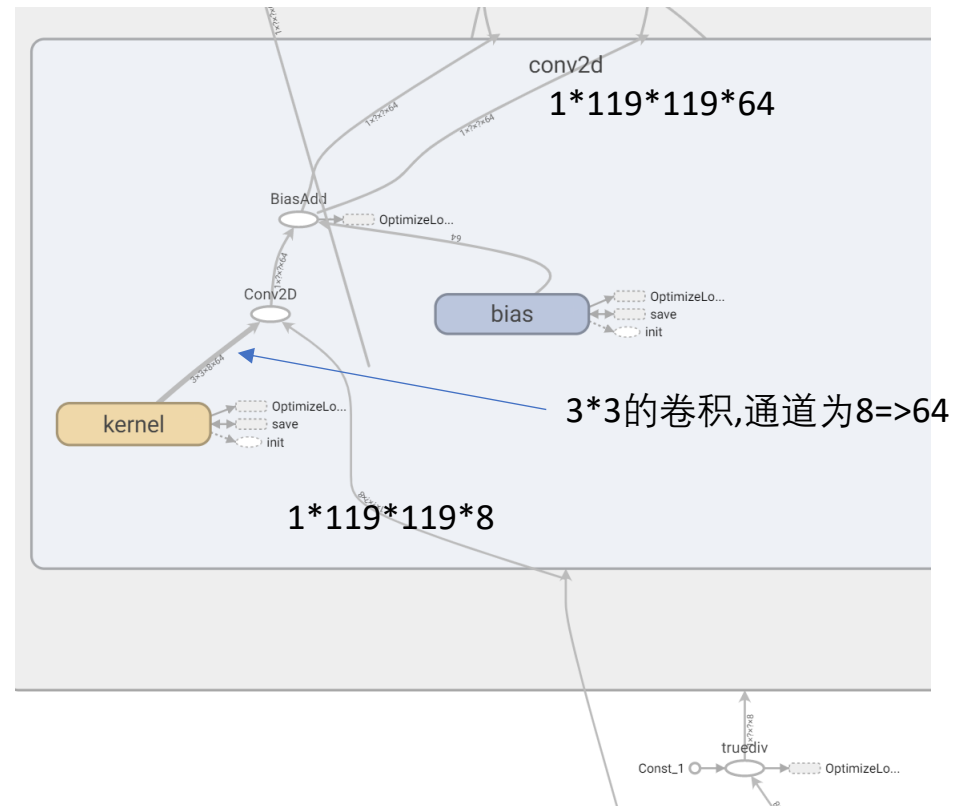


Models can be visualized with *tensorboard*

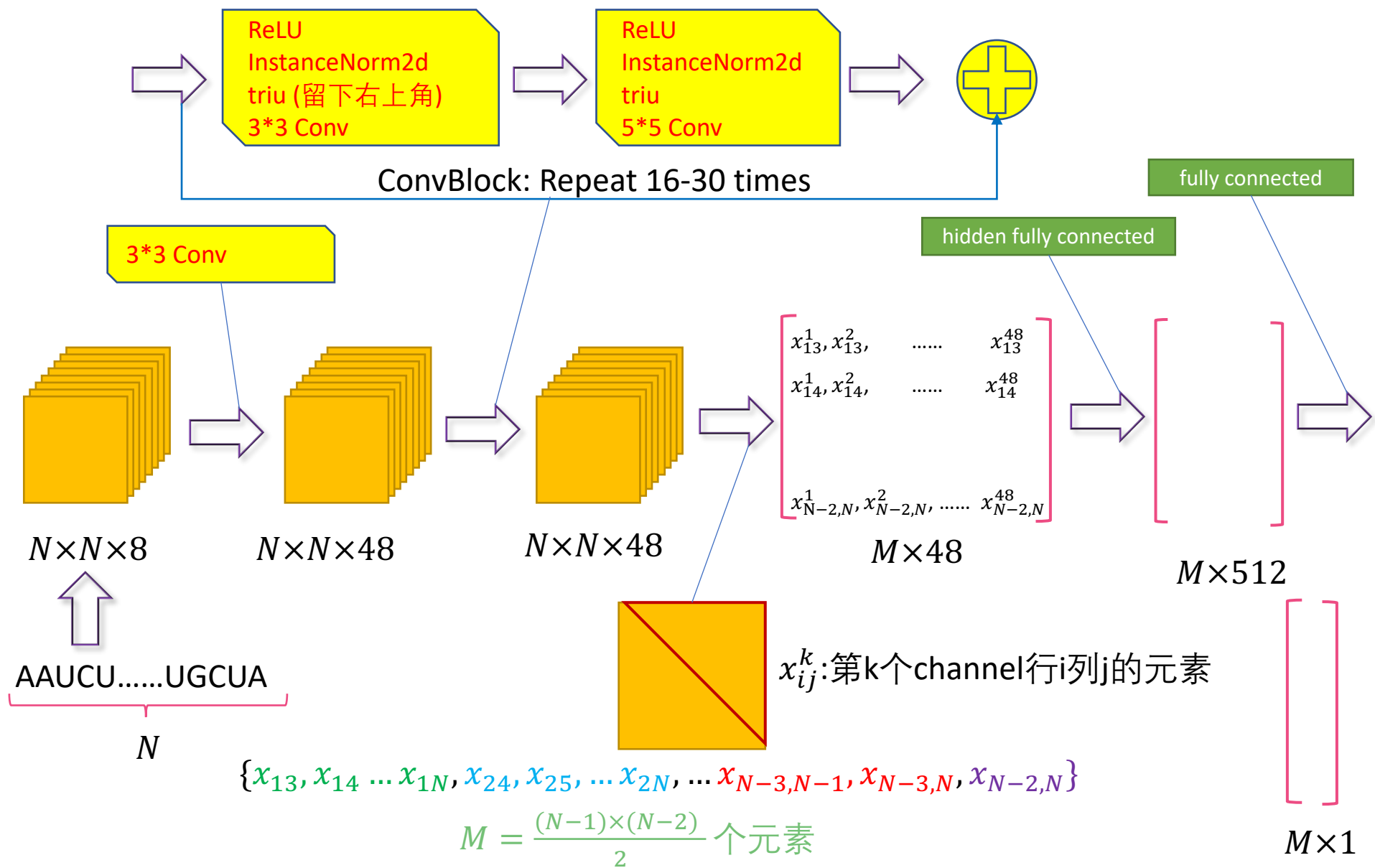
model 0



Derive the dimensions for each pathway



Model structure of SPOT-RNA



A loss function to give more gradient for paired bases (FN samples)

\hat{y} is the input for sigmoid activation function

$$Loss = (1 - y) \cdot \hat{y} + (4 \cdot y + 1) \cdot [relu(-\hat{y}) + \log(1 + e^{-|\hat{y}|})]$$

设 $\xi = \log(1 + e^{-|\hat{y}|}) > 0$

Classes imbalance problem:

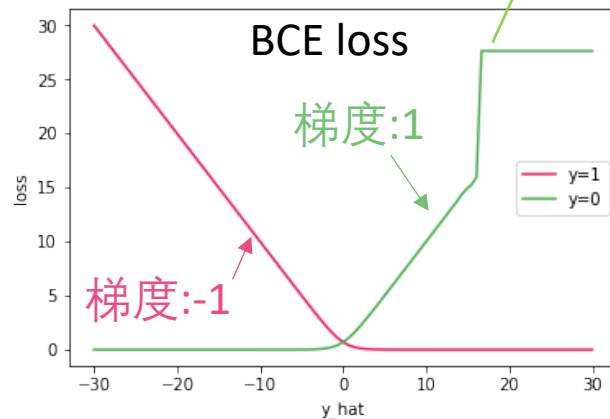
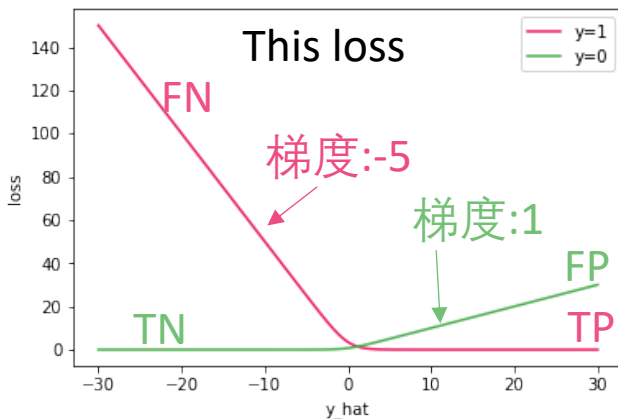
$$\text{Ratio of positive samples} = \frac{\frac{\frac{2}{3}N}{2}}{\frac{(N-1) \times (N-2)}{2}} \approx \frac{2}{3N}$$

$|\hat{y}|$ 越大, 则 ξ 越小

The longer the sequence, the less the proportion of positive samples

underflow

$$\text{梯度} = \frac{\partial L}{\partial \hat{y}}$$



Loss function

$$Loss = (1 - y) \cdot \hat{y} + (4 \cdot y + 1) \cdot [relu(-\hat{y}) + \log(1 + e^{-|\hat{y}|})]$$

\hat{y} 是Final FC输出的sigmoid之前的数

$$\text{设 } \xi = \log(1 + e^{-|\hat{y}|}) > 0$$

$|\hat{y}|$ 越大, 则 ξ 越小

① $y = 1; \hat{y} > 0$

$$Loss = 0 + (4 + 1)[0 + \xi] = 5 \cdot \xi \quad \hat{y} \rightarrow +\infty \text{则Loss越小}$$

② $y = 1; \hat{y} < 0$

$$Loss = 0 + (4 + 1)[- \hat{y} + \xi] = 5 \cdot (-\hat{y} + \xi) \quad \hat{y} \rightarrow 0 \text{则Loss越小}$$

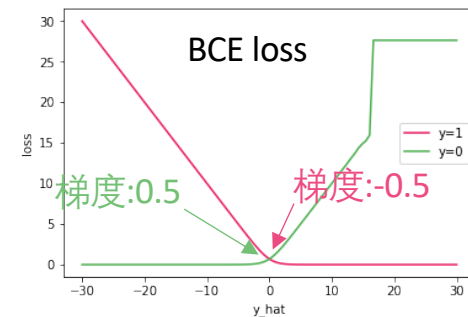
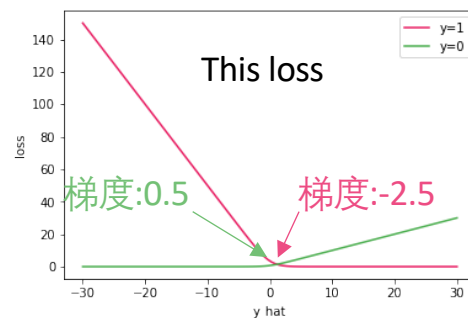
③ $y = 0; \hat{y} > 0$

$$Loss = \hat{y} + 1 \cdot [0 + \xi] = \hat{y} + \xi \quad \hat{y} \rightarrow 0 \text{则Loss越小}$$

④ $y = 0; \hat{y} < 0$

$$Loss = \hat{y} + 1 \cdot [-\hat{y} + \xi] = \xi \quad \hat{y} \rightarrow -\infty \text{则Loss越小}$$

$$\frac{\partial L}{\partial \hat{y}}$$



Parameter configuration of five models

	#convblocks	#channels	α in loss function	LSTM	#Hidden layer
Model 0	16	48	4	✗	2
Model 1	20	64	4	✗	1
Model 2	30	64	4	✗	1
Model 3	30	64	0	✓	0
Model 4	30	64	0	✗	1

$$Loss = (1 - y) \cdot \hat{y} + (\alpha \cdot y + 1) \cdot [relu(-\hat{y}) + \log(1 + e^{-|\hat{y}|})]$$

The larger the alpha, more gradient for **False Negative samples**

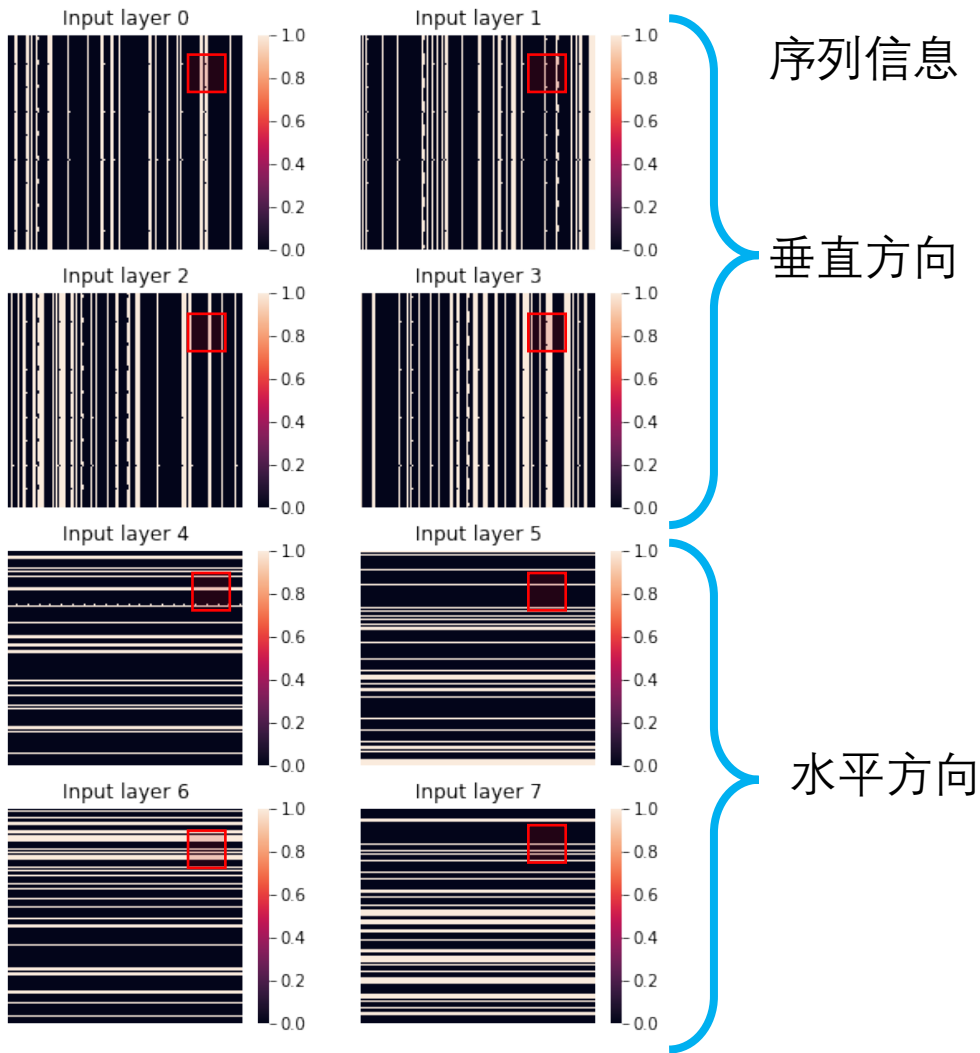
Input layer: Let a convolution get the information of the clip pairing

Example: Channel 3

A U C C G

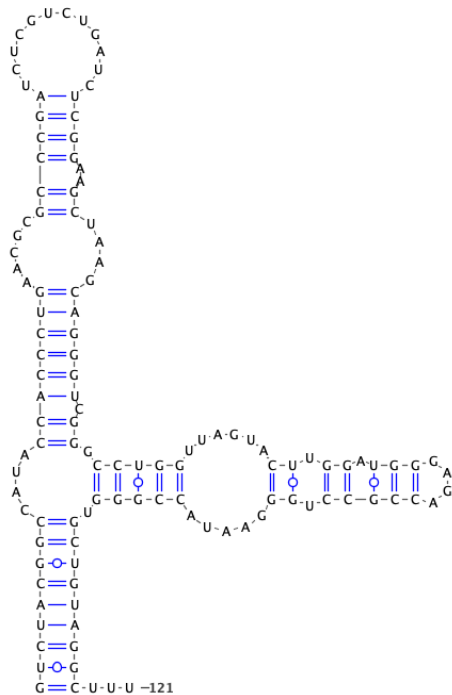
A		0	0	1	1	0
U		0	0	1	1	0
C		0	0	1	1	0
C		0	0	1	1	0
G		0	0	1	1	0

Input: 8 channels



	8 Channels
A-A	10001000
A-U	10000100
A-C	10000010
A-G	10000001
U-A	01001000
U-U	01000100
U-C	01000010
U-G	01000001
.....

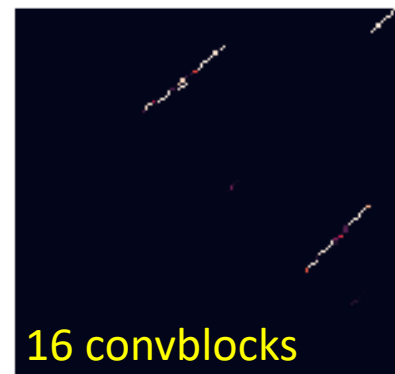
Model output: an intuitive display



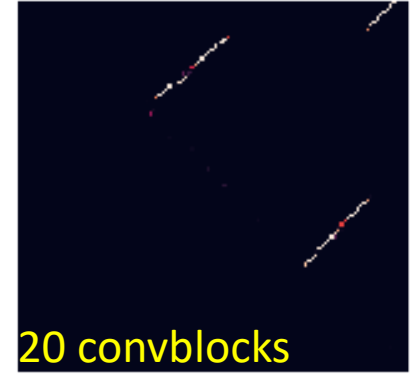
Target



model 0



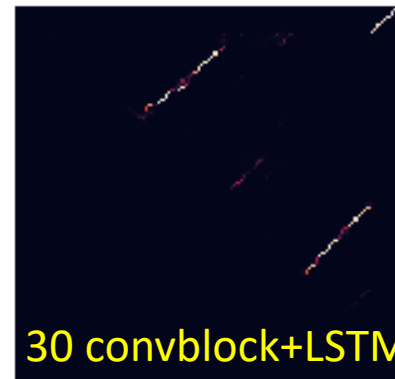
model 1



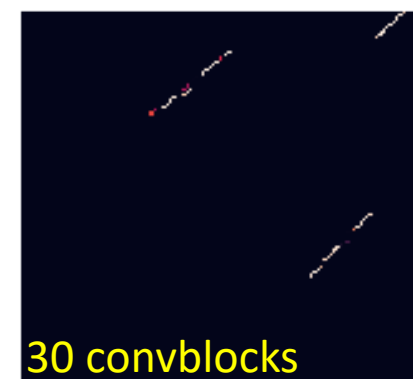
model 2



model 3



model 4



Obtain a valid secondary structure from a pairing probability matrix using a greedy algorithm

Input: Probability matrix $X: L \times L$, L is the length of sequence

Output: Resolved valid secondary structure $S: \{(x_i, x_j) | (x_i, x_j) \in S\}$

Process:

Collect base pairs with probability greater than 0.335

$$P = \{(x_i, x_j) | X(x_i, x_j) \geq 0.335\}$$

Ambiguous base pair list: $ABPlist = \text{get_ambiguous_basepairs}(P)$

while $ABPlist \neq \emptyset$:

for list $l \in ABPlist$:

$$(x_i, x_j) := \underset{i,j}{\operatorname{argmin}} \{X(x_i, x_j) | (x_i, x_j) \in l\}$$

remove (x_i, x_j) from P

$ABPlist = \text{get_ambiguous_basepairs}(P)$

process $\text{get_ambiguous_basepairs_list}(P)$:

1. Ambiguous bases $B := \{x_i | x_i \in P \text{ and } \text{count}(P, x_i) > 1\}$, $\text{count}(P, x_i)$ is the number of times x_i occur in P

2. Ambiguous base pair lists $ABPlist = \text{list}()$

for b in B :

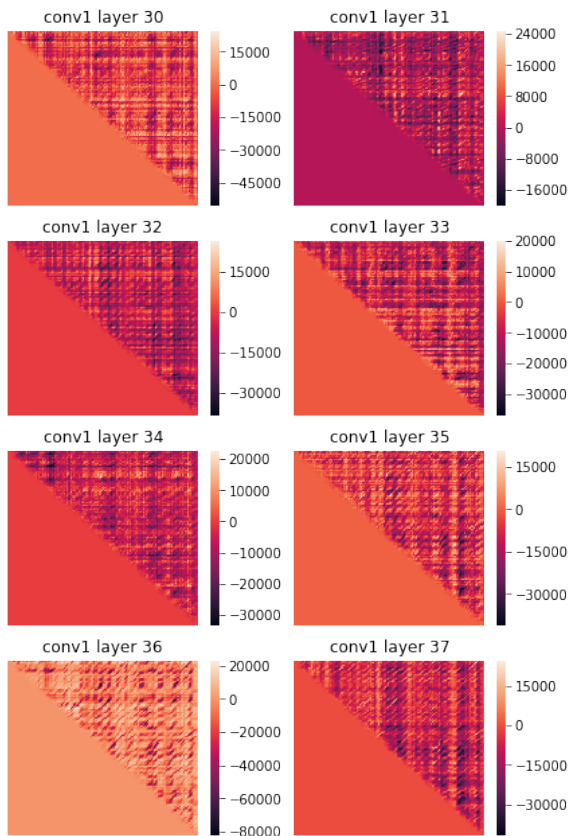
initialize list $l = \text{list}((x_i, x_j) | b \in (x_i, x_j) \text{ for } (x_i, x_j) \text{ in } P)$

$ABPlist.\text{push}(l)$

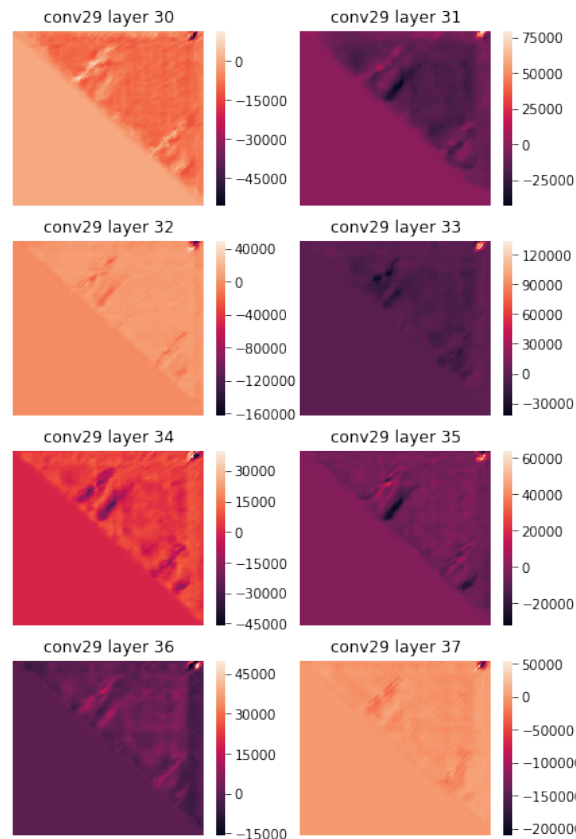
return $ABPlist$

Intermediate layer

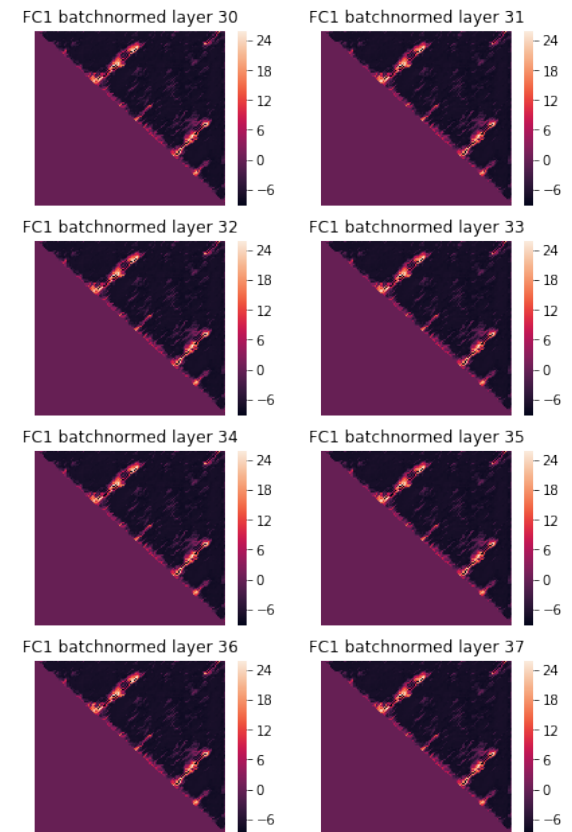
The 1st ConvBlock



The last ConvBlock



Fully connected layer



My implementation

15 ConvBlock + 2 hidden FC

48 Channels

Dropout=0.3

weight_decay=0

Adam optimizer, lr=0.003

50 Epochs

training data: bpRNA training

Metrics

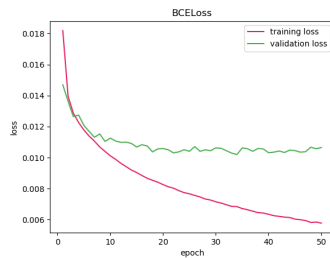
$$accuracy = \frac{TP + TN}{Total}$$

$$precision = \frac{TP}{TP + FP}$$

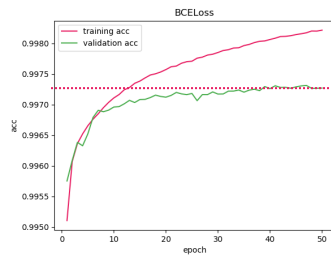
$$recall = \frac{TP}{TP + FN}$$

$$F1\ score = 2 \times \frac{Precision * Recall}{Precision + Recall}$$

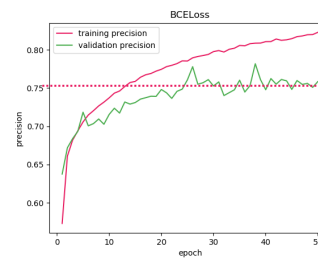
loss



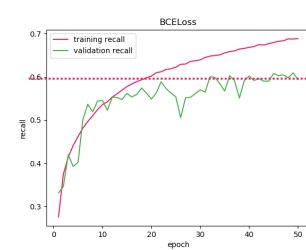
acc



precision



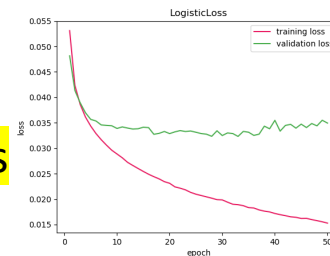
recall



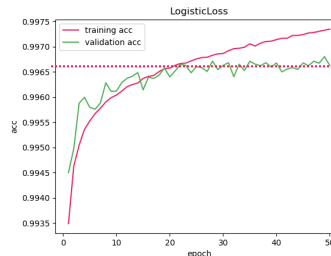
(test set)

preci=0.772
recall=0.606
F1=0.679

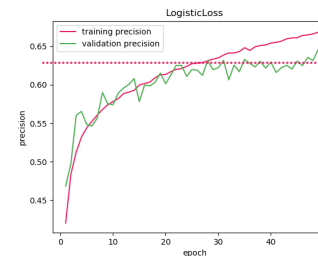
Logistic Loss



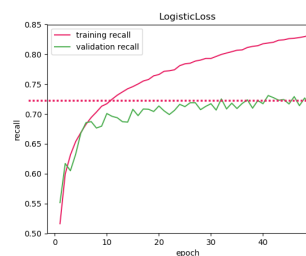
Logistic Loss



Logistic Loss



Logistic Loss



preci=0.632
recall=0.734
F1=0.679

BCE Loss

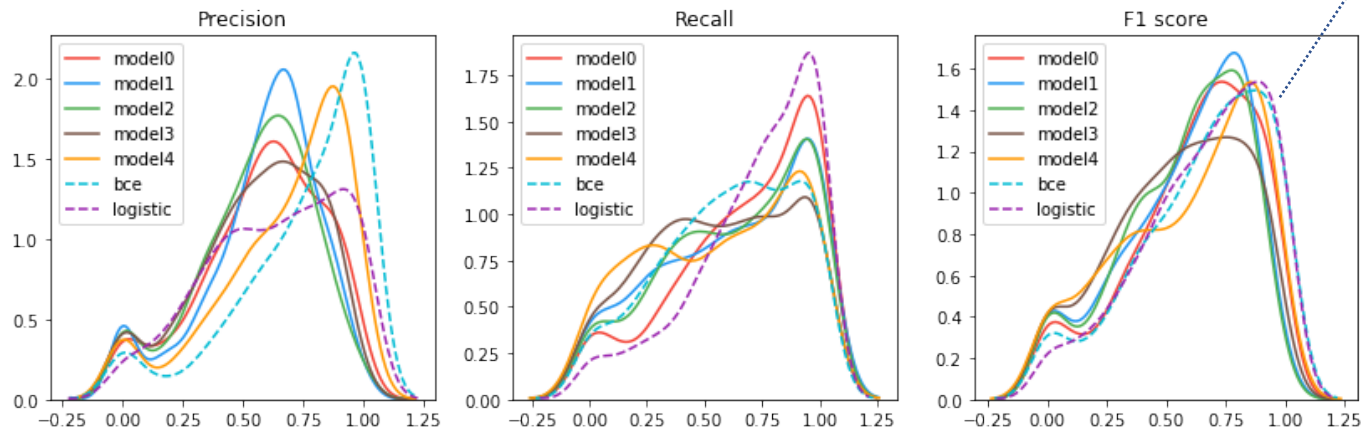
Logistic Loss

Systematic comparison of performance of these models

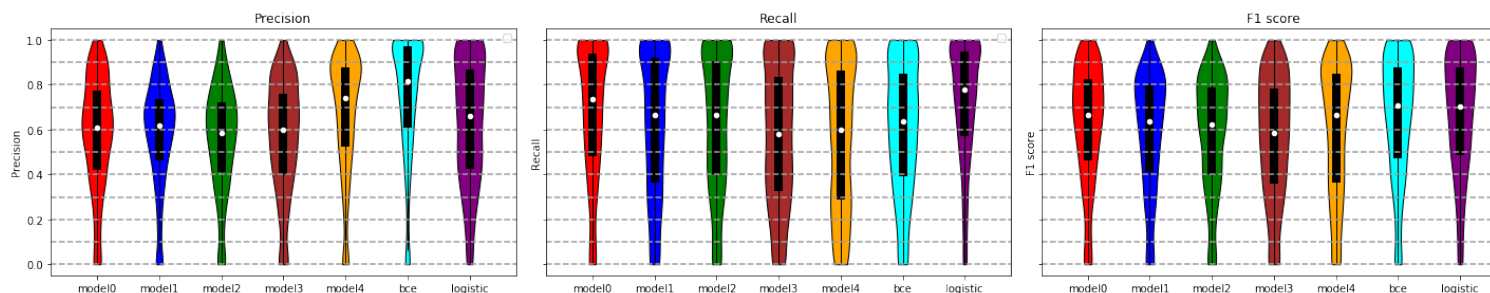
On bpRNA validation set

Maybe transfer learning changes the model's preferences

Distribution plot



Violin plot

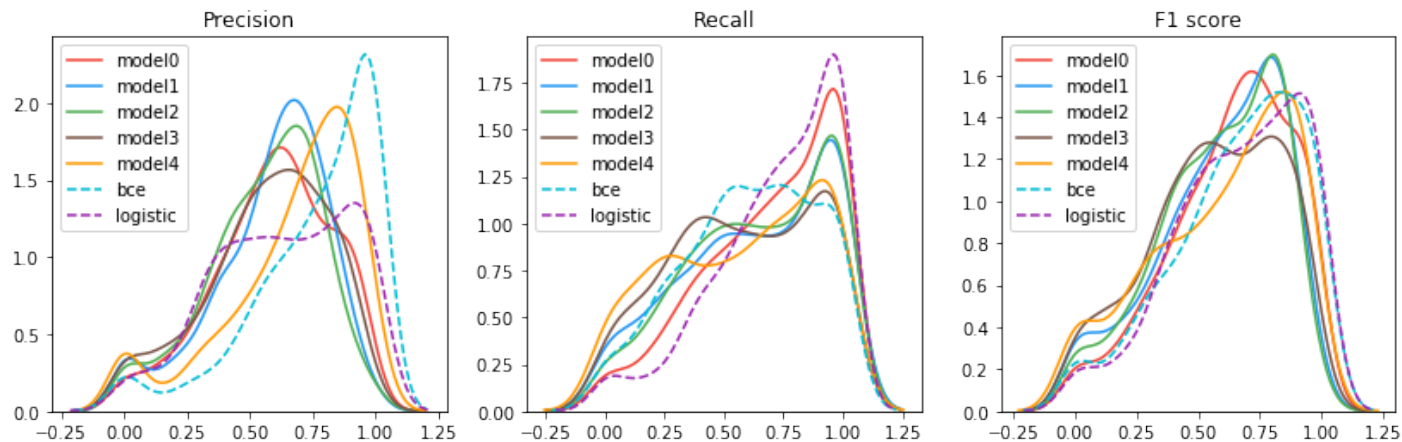


- Larger precision for BCE model; larger recall for BCE model
- My implement models have slightly higher F1 score than model 0

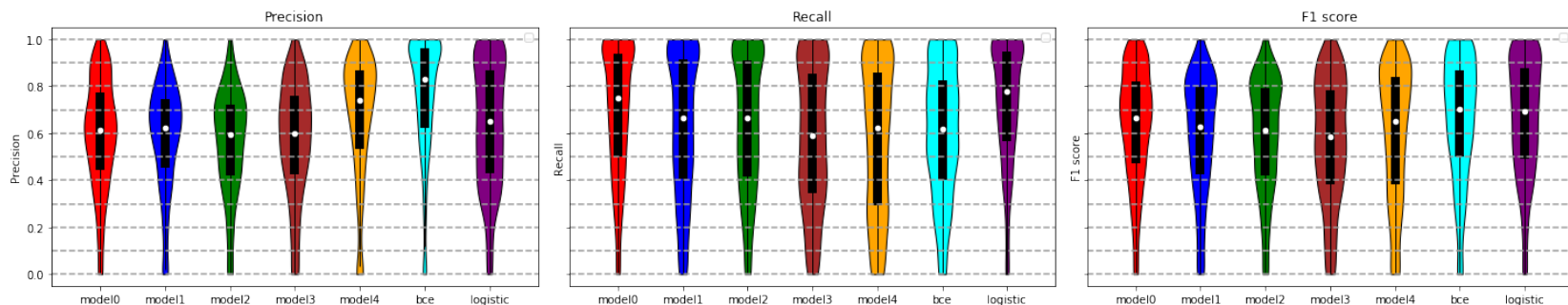
Systematic comparison of performance of these models (Cont'd)

On bpRNA test set

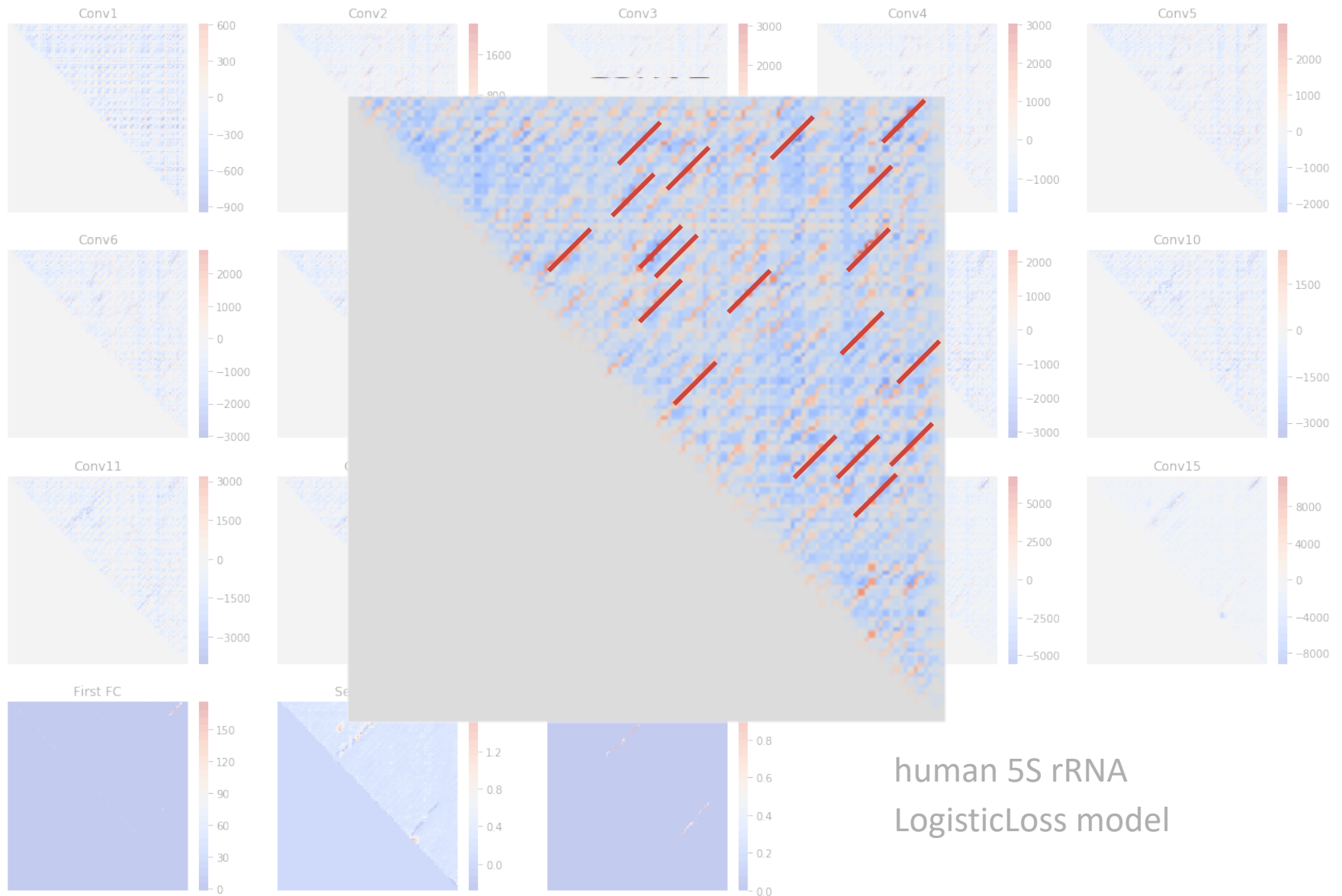
Distribution plot



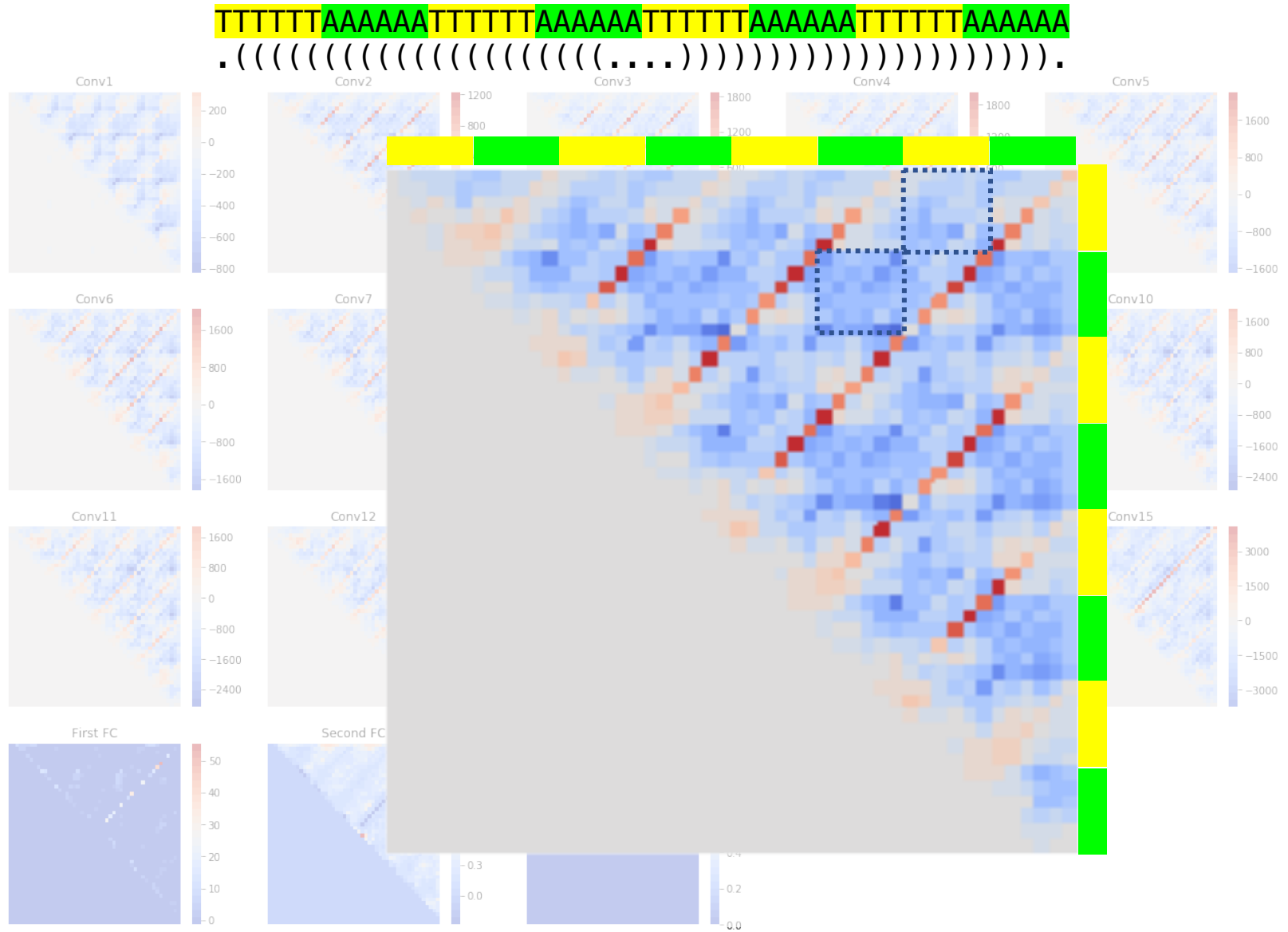
Violin plot



What did the model learn?

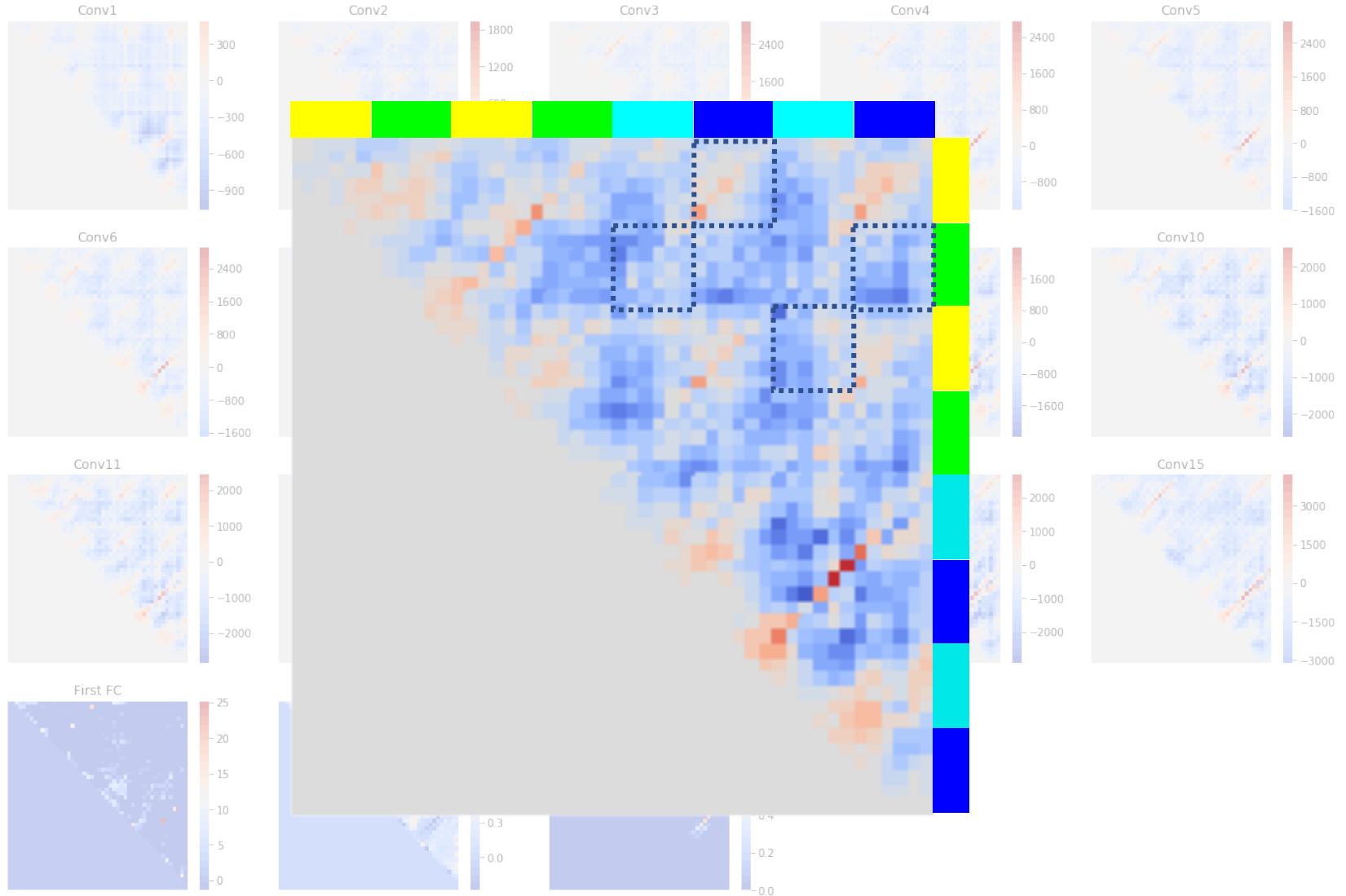


Convblocks learn local pairing rules; FC learn to maximize base pairs (Cont'd)



Convblocks learn local pairing rules; FC learn to maximize base pairs (Cont'd)

TTTTTTAAAAAATTTTTAAAAACCCCCGGGGGGCCCCCGGGGGG
((((((((((.....))))))))(((((.....))))))))



SPOT-RNA highlights

- Learning the pairing pattern from known secondary structures
- Can predict psuedoknots
- Can predict non-canonical base-pairing

SPOT-RNA defects

- Limited by training data sets: length bias (≤ 500), RNA type bias, data quality, data size, ...
- Only one structure produced for one sequence: unable to predict structure under different conditions; unable to predict structure ensemble.

So, integrating experimental data to predict RNA secondary structure can greatly improve the scope of application of the model

Learning ensemble base pairing probability

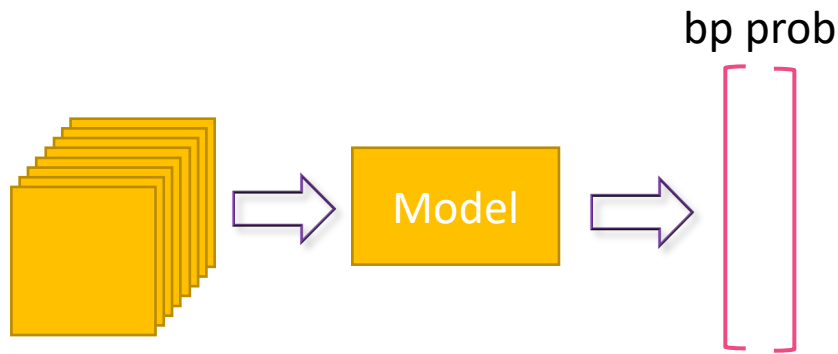
A ConvBlock + B hidden FC
C Channels
Dropout=0.3
weight_decay=0
Adam optimizer, lr=0.001
100 Epochs

training data: bpRNA training

Leaving the model that minimizes the MAE of the **validation** set

testing data: bpRNA testing

Mean absolute error (MAE) = $\frac{1}{N} \sum_{i=1}^N |\hat{y}_i - y_i|$



Loss/A/B/C	testing MAE
MSELoss_15_2_48	0.17030
MSELoss_25_2_64	0.16358(60 epochs)
BCELoss_15_2_48	0.17058
MSELoss_15_2_64	0.16009

MAE=~0.16 is too large!

What to do next?

1. Make model more complex (more layers, more channels...)
2. Learning pairing probability is too hard for model, so give more input (DP middle matrix) to model.
3. Use other training data
4. ...

Can we integrate SCFG to deep learning?

Stochastic Context-Free Grammar



$$\begin{aligned} s &\rightarrow C s_1 G \\ s_1 &\rightarrow A s_s U \\ s_2 &\rightarrow b_1 b_2 b_3 \\ b_1 &\rightarrow U \\ b_2 &\rightarrow U \\ b_3 &\rightarrow C \end{aligned}$$

Each deviation can be assigned a prob
(Stochastic grammar)

- SCFG is usually used in homologous structure modelling, this requires multiple **similar** sequences to be aligned
- But we need to predict the structure of a sequence without any **prior knowledge**

Base on this model (pairing probability)

How to integrate?

Reinforcement learning

Problem:

1. action space is too large;
2. different environment for different RNA