CSEP 527 HW 5- MM Report

Isabelle Lee, (ID: 1423762)

December 7, 2020

1 How to run

```
python src/train_model.py -longl 1400
                          -shortl 50
                          -k 5
                          -pseudo 1
```

Note: without any parameters (all parameters are optional), it defaults to the set of default parameters given with long length defined as 1400, short length defined as 50, k window defaults to 5, and pseudocount defaults to 1.

For the plots in extra credit section, please run

```
python src/extra.py
```

2 Training Markov Model on Default Set of Parameters

Below is the output of the `train_model.py` script:

```
total number of orfs found: 106819
number of long orfs: 118
number of short orfs: 81737

reading frame 3
 total number:  35200
  first:  (0, 36)  (length of 36)
  last:  (1664967, 1664970)  (length of 3)

reading frame 1
 total number:  35933
  first:  (1, 94)  (length of 93)
  last:  (1664920, 1664970)  (length of 50)

reading frame 2
 total number:  35686
  first:  (2, 5)  (length of 3)
  last:  (1664963, 1664970)  (length of 7)

total number of CDS strands:  848
shortest orfs:
       start  end  length     score  matches
0          0   36      36  6.634133    False
71133      2    5       3  0.000000    False
```

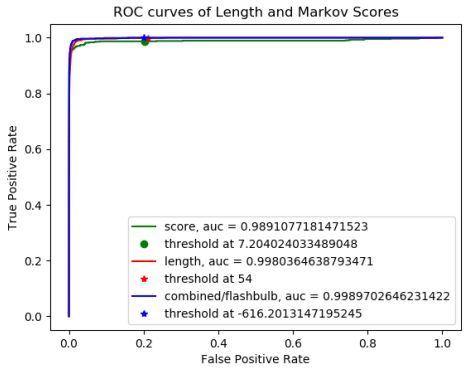
```
71134       8    20       12   0.720851     False
71135      23    32        9  -0.451101     False
1          39    51       12   2.752885     False
longest orfs:
        start     end  length       score  matches
71526   17618   19229    1611  608.387880     True
36031   33625   35245    1620  835.974865     True
36169   42724   45109    2385  936.277938     True
72661   74591   76010    1419  503.758744     True
36888   76819   78481    1662  715.488367     True

ORFs Found: total number of orfs found: 106819
number of long orfs: 118
number of short orfs: 81737

P count(AAGxyT):
      A    C     G     T
A   307   51   223   394
C   104   15    12   119
G   211   42    39   218
T   148   19    68   198
Q count(AAGxyT):
      A    C    G     T
A    90   26   48    95
C    87   22   15   119
G    41   20   26    59
T   139   39   64   175
```

I used normalization to make sure the conditional probabilities add up to 1. So, the MLE approximation equation looks like

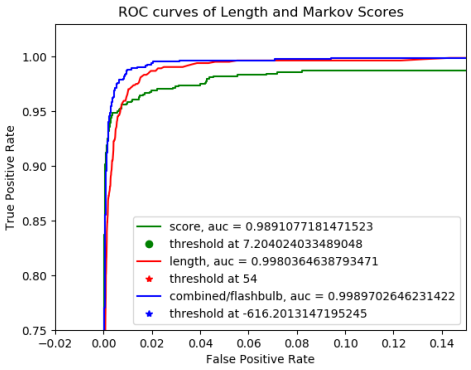
$$P(w_i|w_{i-1-k}^{i-1}) = \frac{Count(w_{i-1-k}^{i}) + p}{Count(w_{i-1-k}^{i-1}) + pV} \tag{1}$$

where p denotes pseudocount and V denotes total unique counts of k-mers.

From looking at this report, the longer ORFs of length greater than 1400 is very likely to be an actual gene annotated in CDS file. This is also reflected in the Markov model scores. The model scores were higher for longer ORFs than shorter ORFs. Then, we can immediately pick out two features that could represent the dataset well: length of ORF and the Markov model scores (defined as the log of foreground and background probabilities of the ORF sequence).

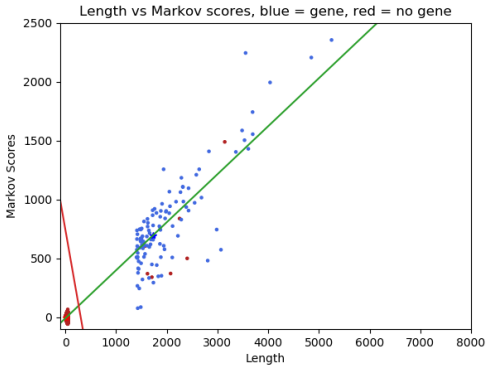


(a) ROC plot based on Markov Scores, length, and Flashbulb combined

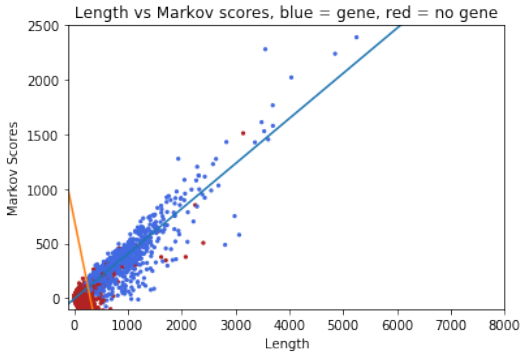


(b) Zoomed in

Figure 1: ROC curves with default parameters given in assignment



(a) Flashbulb method to combine length and scores



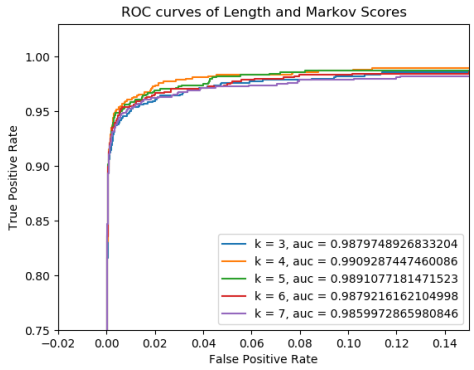
(b) With full data

Figure 2: Combining two features

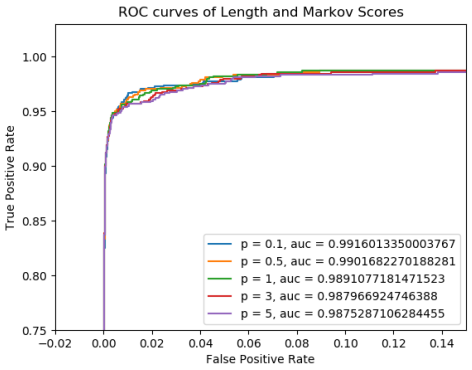
From the ROC plots, the scores and lengths separately are good features indeed. The score curve in green crosses the length curve in red as seen in the zoomed view. The AUCs are listed according to the plot as well. Initially, the Markov scores achieve higher TPR with less FPR, but the length plot crosses over as having a better tradeoff. In order to utilize both of these features, Flashbulb method outlined in the assignment was implemented. Essentially, this method draws a decision boundary between the length vs. Markov score plane. Then, the distance from this boundary can be a good combination of the two feature for their tradeoff. The blue curve appears to achieve this well, as it envelops the two score and length ROCs, with the highest AUC.

3 Extra credit: Hyperparameter Tuning

For extra credit, I explored hyperparameter tuning, with varying pseudo-counts and k window length, long ORF cutoff and short ORF cutoff. These variables are chosen one by one and varied by a range of values, with other variables held at the default parameter values. The results are summarized as ROC curves in Figure 3. Based on AUCs from these variables, the best hyperparameters seem to be long ORF cutoff of 800, k-mer window size of 4, pseudo-count of 0.1, and short ORF cutoff doesn't seem to affect the model much.

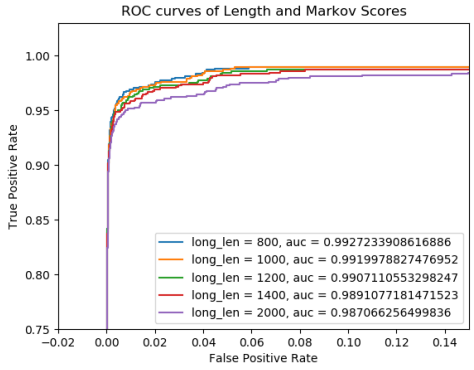


(a) varying k-mer window size

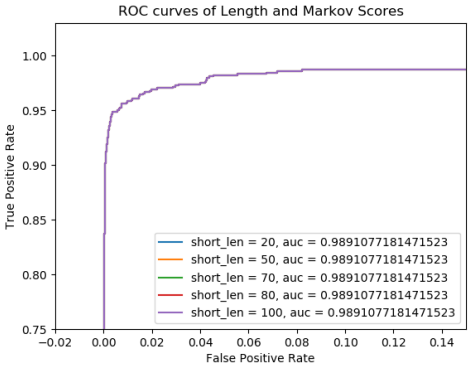


(b) varying pseudo counts

Figure 3: ROC curves with varying k values and pseudo values



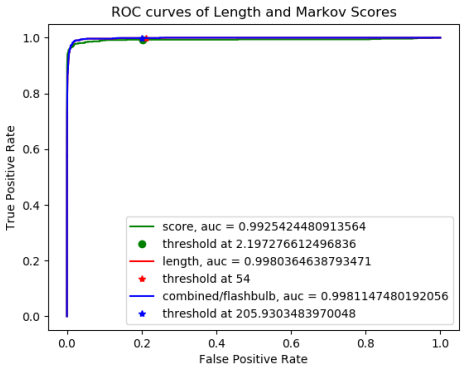
(a) varying long ORF cutoff



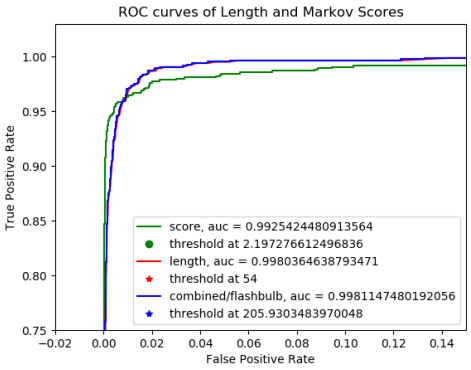
(b) varying short ORF cutoff

Figure 4: ROC curves with varying ORF lengths

With these hyperparameters chosen, we can re-train the Markov model and combine the resulting scores with length using Flashbulb method. The results are reported in Figure 5. The AUCs for Markov score ROC has improved from the baseline model in Section 2. The length ROC is largely unaffected by the hyperparameter changes. More glaringly, the combined method no longer yields higher accuracy. Perhaps this is because the length data and Markov score data start to overlap significantly more, as we include shorter ORFs with smaller k-mer window size, such that a linear boundary is no longer effective. I'm running out of time, so I haven't completed a new decision boundary. However, I think it would be interesting to try something like a quadratic boundary or hyperbolic boundaries to wrap around the long ORF positive points more neatly, with the location of vertex moveable between long data points and short data points. If I had even more time, I think it would be good to try sklearn's SVMs for softer boundaries too. Thanks for an awesome class!



(a) ROC with chosen hyperparameters (k=4, pseudo = 0.1, long ORF cutoff at 800)



(b) Zoomed in

Figure 5: ROC curves with varying ORF lengths