Python exercises for the lecture "Models and Algorithms in Bioinformatics"

Organizers: Peter Meinicke, Sina Garazhian
Submission date: 16.11.2024, 12 p.m.

Sheet 1: Position Weight Matrices (PWM) for start codon detection

Implement the identification of start codons as presented in the lecture using position weight matrices. The PWM model should represent a region of L bases just before the start codon, a so-called Translation Initiation Site (TIS). The sequences for estimating the model parameters (emission probabilities) can be found in the text file `TIS-Ecoli.txt`.

Each line in the file corresponds to a sequence section of length 200, in which the beginning of the start codon is in the middle at position 101. Possible start codons that we consider here, are **ATG**, **GTG**, **TTG**. How many possible candidates for the different start codon variants can be found in the sequences? First count the actual "correct" (true) start codons in the middle, then count the "wrong" (false) ones, i.e. candidates that can be found in coding (right side) and non- coding area (left side). Do all sequences show a valid true start codon? Do you see a difference in the distribution of codon variants between *true* and *false* ones?

First use a PWM length of $L = 30$ and the uniform distribution as background model, i.e. the background probabilities are 0.25 for all bases at all positions. Use a pseudocount of $r = 1$ when estimating the probabilities. Consider only candidates where all L positions are observable. Try to convert the sequences to numpy vectors so that you can create binary (boolean) matrices to finally count all positional (column) base frequencies simultaneously with iterative matrix summation. Hint: use the equality operator (e.g. **seq_vec == 'A'**) to build boolean matrices with four rows indicating the different bases occurring at the corresponding positions.

Now try to find a suitable detection threshold for the logarithmic score. The score is generally only calculated for valid start codon candidates with all L required positions observable (see above).

First select a minimal threshold t such that at least 50% of the correct candidates are recognized, according to a score larger than t. So we achieve a 50% true positive rate (TPR). How high is the error, i.e. how many of the wrong candidates with score above t do you inevitably detect too? Calculate and report the false positive rate (FPR).

Now divide the data into a training and a test set. Use the first 400 sequences to estimate the parameters and determine the detection threshold according to the 50% TPR criterion. Use the rest of the sequences to estimate the detection rate (test TPR). What is the FPR test-error here?

With the ROC curve, the detection accuracy of the method can be determined independently of a specific threshold (see lecture). Plot the curve (function plot) and determine the AUC (area under curve) value.

Try to improve the AUC value on the test data by various modifications. First, shift the PWM window one position into the coding region, i.e. you are now using 29 positions before the start codon and the first base of the start codon. Does this increase the

accuracy? What could be seen as a disadvantage of this extension with regard to detection of the different start codon variants?

Also try alternative values for the pseudo-count parameter. Do lower or higher values according to $r = 0.1, 0.5, 2.0, 10.0$ lead to an improvement? What possibilities are there to improve the background model? Try to estimate a complete background model with L positions from all valid false candidates in the training set (with all L positions observable). The estimation scheme should be the same as for the true model before. For the possibly improved model, report the ROC curve and the AUC value.

Try to find information about gene prediction in prokaryotes, in general. What could be done for further refinement or extension of the basic setup that we consider here? Write down your ideas but do not implement them. Just think beyond the possibilities examined here in our restricted setup! In this setup, we assumed all start codon candidates on the left or right side of the actual start codon in the middle to be false candidates. Is this assumption always correct? What should an improved setup theoretically look like that also takes stop codons into account?

Submission instructions

Please submit your solution by **16th November, 12 p.m.** using the corresponding homework folder in StudIP. Upload a **zip** compressed file containing all documents and data that you use in your solution. The file name should indicate the author, for example *Peter_Meinicke_Sheet1.zip*.
With your solution you should provide the Python source code together with the documentation within a **Jupyter Notebook** answering all the questions above. Use Markdown cells for your answers and explanations well separated from the Python code.
For the programming in Python/numpy do not use any code or functions from special bioinformatics libraries or toolboxes! Make sure that your code is readable and understandable for others, i.e. use descriptive names for the variable definitions according to the conventions on **Sheet 0**, do not use implicit variables (hard-coded values without a variable) and make use of comments!