We have to be more organized about comp bio project. It is worth 50 marks!! [More than any other individual project]

Please follow the extension conventions. It will save us a lot of work when we will have to integrate everything(believe me).

These are the tasks that must be completed--------------------->

Checklist:

**0)** Basic task. Now that things are working properly, this is a basic step which we should complete>>

**Divide the entire raw files into three different folders. First folder should be named "training\_set"(nearly 60% will go here), second folder should be named "testing\_set"(nearly 20% will go here) and the final folder should be named "validation\_set"(nearly 20% will go here).**

Because we have to do cross validation and tweak the \lambda parameters, all this must be done in the validation\_set to avoid over-fitting.

**1) Script for getting fasta from corresponding fast five file[only the template strand].**

I/P: <file\_name>.fast5

O/P: <file\_name>.fasta

**---Completed**

**2) Script(or program) for getting event data from the corresponding fast five file.**

I/P: <file\_name>.fast5

O/P: <file\_name>.events

**---Completed**

**3) Script to read each fasta and compare it with the original genome and save the blast output.**

I/P: <file\_name>.fasta

O/P: <file\_name>.blast

**---Completed(by Srikant I think)**

**4) Parsing the blast output and storing it to a file.**

I/P: <file\_name>.blast

O/P: <file\_name>.alignedfasta

Store it in the following format:

<starting\_point> <ending\_point>

<aligned fasta string>\n

<original fasta string>

**-- Done already - Srikant**

-----Now the harder parts------

**SVM Multiclass:**

**5) C++ Program to generate feature vector for SVM using aligned fasta and events file.**

I/P: <file\_name>.alignedfasta && <file\_name>.events

O/P: <file\_name>.svm

**--- Almost done by me. Has some weird edge case errors(like the case when there are multiple consecutive base pairs). Also, does not take into account reverse strands, and hence will have to change to code for that.**

**6) Write script for running the SVM train command.**

I/P: a set of files(folder) in .svm format

O/P: <model\_file>.model

(the model description)

**7) Write a script for running the SVM test command on the validation set.**

I/P: the validation folder(a set of alignedfasta files)

O/P: a set of .predictedfasta files.

**8) Cross-validate using different hyper parameters and select the one which minimizes the errors. I do not think that using F1 score would be a good testing parameter since we are not using perfect training sets. Hence, edit distance of our predicted alignment vs the actual alignment might be a better error metric. Will confirm with Prof in class tomorrow.**

**9) Write a program for calculating the edit distance between alignedfasta and predictedfasta.**

I/P: <file\_name>.alignedfasta and <file\_name>.predictedfasta

O/P: Edit\_distance between the two fasta files.

**-Done (Srikant)**

**10) Write a script that calculate and prints the net error(Average error). We might have to come up with some multiclass equivalent of F1 score.**

I/P: a set of .alignedfasta and .predictedfasta files and the edit\_distance program.

O/P: Average edit distance and some equivalent measure of F1 score if we are able to define it.

**NAIVE BAYES**

**11) Write a program to convert the .alignedfasta and .events file into a feature vector form that is suitable for Naive Bayes(we can use the same one as SVM or change it according to what is easier)**

I/P: <file\_name>.alignedfasta and <file\_name>.events

O/P: <file\_name>.svm/<file\_name>.naive

Simultaneously create and output folder in Naive Bayes format as well.

**12) Write a program to calculate the different required probabilities from the training set and write these in a file.(make classes of the input signals depending on the range)**

Note: It will be better to just store the counts rather than the actual probability since there might be floating precision problems while reading from the file and writing it in a buffer. Better to just store the numbers and calculate probability on the go. The main problem with our project is scale and we have to think accordingly.

I/P: the training folder

O/P: <file\_name>.model …. stores the model in some defined format.

13) Write a program to make the predictions using naive bayes.

I/P: a set of .

**14) Write scripts for cross validating and testing. There are no real hyperparameters here. So we might skip the cross validation altogether and directly train on the training and validation set and test on the testing set.**

**BASELINE**

We might have to design a baseline model as well. Thats what Vivek said in the feedback. We will have to ask professor Patro and confirm it.