Full name : Saad Benslimane
ID : 20228273
Email adress : saad.benslimane@umontreal.ca
March 3th, 2022

Instructions

- *Find the practical assignment in the form of a Jupyter notebooks,* `main-EN.ipynb` *(English version) or* `main-FR.ipynb` *(French version), in Github repository. The repo includes everything you will need except the data.*
- The data can be found in this Google Drive link.
- *For questions 1,2,3 and 4, you will have to fill parts of the code in the file* `solution.py` *(from the Github repo). This is the file that should be uploaded to the course GradeScope for auto-grading.*
- *Work off the template and DO NOT modify the name of the file* `solution.py` *since the code will be automatically graded.* It is very important that you only add code in the specified locations and do not modify the designated input or outputs!
- *Your solution to question 5 should be written as a LaTeX document (no longer than one page). You can use this latex as a template for that.*
- ***TAs for this assignment are Matthew Scicluna and Akram Erraqabi.***

Once you have located the the Github repo and the Google Drive folder, you can begin the assignment. We will provide the instructions to run the notebook in Google Colab, but you should be able to adapt it on whatever computing platform you would like to use.

1. Using the trained Basset model (learning rate=0.002, batch size=64, number of epochs=5) and the untrained Basset model, we notice that the accuracy of the model increased after training. As we can see from (1) and (2) the trained model is working more like the smart model we implemented before with an AUC value of 0.8429 (0.94 for smart model), while in the other hand the untrained model is more like the dumb model with an AUC value of 0.4975 (0.4641 for dumb model).

2. Before reproducing what was done in the article, we should normalize the matrix $CTCF$ so that each column totals 1. This can be checked from (3) the three extracted columns from the normalized matrix satisfied the condition.

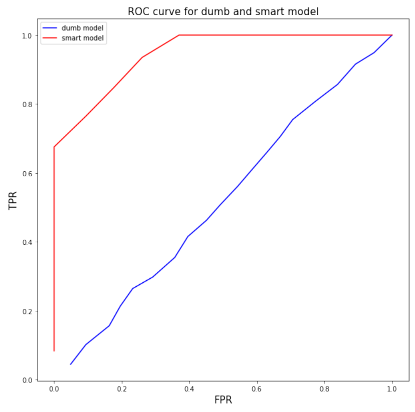

Figure 1: ROC curve for smart and dumb model

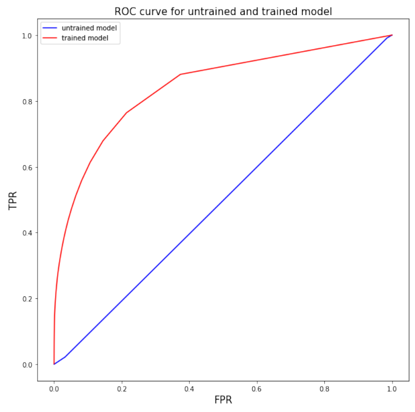

Figure 2: ROC curve for trained and untrained model

$$\begin{pmatrix} 0.0952 & 0.1829 & \cdots & 0.4427 \\ 0.3187 & 0.1588 & \cdots & 0.1993 \\ 0.0832 & 0.4534 & \cdots & 0.2929 \\ 0.5027 & 0.2048 & \cdots & 0.0649 \end{pmatrix}$$

Figure 3: Normalized CTCF matrix

3. The next step is to find the maximum value enabled for each of the 300 filters. This was done by passing the test data through the first convolutional layer, and for each filter we take the maximum value of the output as the activation value. From (4) we can see the distribution of the 300 maximum values, the majority of these values are between 17 and 23, with a median of 18.29.

4. As mentioned in the article we did apply the unfold function to the first 10 batches but the correlation between the resulting $PWMs$ and the $CTCF$ matrix was not interesting (maximum value 0.3). By reducing the number of batches to one we see that the correlation increased to 0.597, but when using a higher learning rate (equals to 0.01) this helped us to find a better results with a higher correlation equals to 0.679. From (5) we can visualize the correlation between $CTCF$ and the best 3 PWM_i predictions.

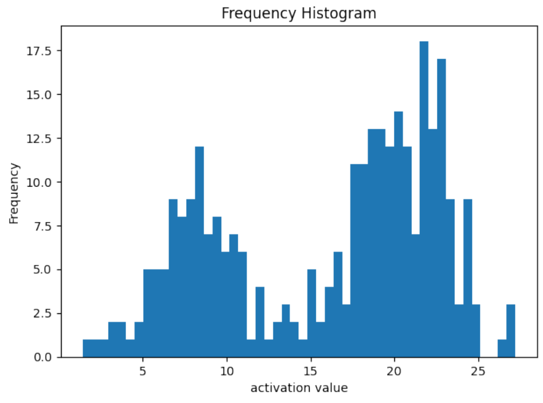

Figure 4: Distribution of maximum activated values

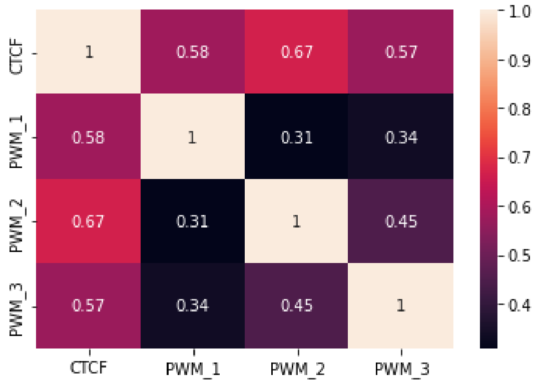

Figure 5: Heatmap for CTCF and the three best PWM