LAB 5: deep learning for genome classification

ASSIGNMENTS

First, take a look at the **LAB5_Tips** file and practice with each section described. We strongly believe that they could be helpful with the following assignments.

Assignment 1: CNN to predict intron/exon boundaries

With this assignment we will practice with a Convolutional Neural Network for the classification of DNA sequences, in particular for the prediction of the intron exon boundaries. Download the **splice.data** file from the Teaching Portal and proceed with the following steps.

1. Read the file and store the DNA sequences with the respective labels ("EI" for the exon-intron boundary, "IE" for the intron-exon boundary, "N" for no boundary)
2. Perform the one-hot encoding of the labels and split the dataset into train and test set
3. Perform the one hot encoding of the train and test datasets to feed the neural network
4. Create and compile a CNN with the following architecture:
    - Convolutional layer with kernel size = 3 and activation function equal to ReLu
    - Maxpooling layer with pooling size = 2
    - Convolutional layer with kernel size = 3 and activation function equal to ReLu
    - Maxpooling layer with pooling size = 2
    - Dense layer with ReLu activation function and 100 units
    - Dense layer with softmax activation function and 3 units

    Start training with the following parameters: learning rate equal to 0.01 and number of epochs equal to 10 (usually a number of epochs between 50 and 100 is used, however, since we have a simple classification problem, in order not to waste too much time you can set this number equal to 10).
5. Predict and evaluate CNN performance onto test set
6. Create final confusion matrix
7. Try other network architectures and/or other training hyper-parameters (e.g. change learning rate or add more convolutional layers and/or Dense layers, kernel sizes, number of units for each layer…)

Assignment 2: LSTM to predict intron/exon boundaries

With this assignment we will practice with a Long Short Term Memory network (**LSTM**) for the classification of DNA sequences, in particular for the prediction of the intron exon boundaries. Download the **splice.data** file from the Teaching Portal and proceed with the following steps.

1. Read the file and store the DNA sequences with the respective labels ("EI" for the exon-intron boundary, "IE" for the intron-exon boundary, "N" for no boundary)
2. Perform the one-hot encoding of the labels and split the dataset into train and test set
3. Perform the one hot encoding of the train and test datasets to feed the neural network
4. Create and compile a LSTM with the following architecture:

- LSTM layer with 50 units followed by a dropout regularizer equal to 0.2
- LSTM layer with 50 units followed by a dropout regularizer equal to 0.2
- LSTM layer with 50 units followed by a dropout regularizer equal to 0.2
- LSTM layer with 50 units followed by a dropout regularizer equal to 0.2
- Dense layer with softmax activation function and 3 units

Start training with a number of epochs equal to 10 (usually a number of epochs between 50 and 100 is used, however, since we have a simple classification problem, in order not to waste too much time you can set this number equal to 10).

5. Predict and evaluate LSTM performance onto test set
6. Create final confusion matrix
7. Create and compile a **Bidirectional LSTM** with the following architecture:

- Bidirectional LSTM layer with 50 units followed by a dropout regularizer equal to 0.2
- Bidirectional LSTM layer with 50 units followed by a dropout regularizer equal to 0.2
- Bidirectional LSTM layer with 50 units followed by a dropout regularizer equal to 0.2
- Bidirectional LSTM layer with 50 units followed by a dropout regularizer equal to 0.2
- Dense layer with softmax activation function and 3 units

Start training with a number of epochs equal to 10 (usually a number of epochs between 50 and 100 is used, however, since we have a simple classification problem, in order not to waste too much time you can set this number equal to 10).

8. Predict and evaluate Bidirectional LSTM performance onto test set and create final confusion matrix
9. Which differences you notice between standard LSTM and Bidirectional LSTM in terms of performances?
10. Try other network architectures and/or other training hyper-parameters (e.g. add more LSTM/bidirectional LSTM layers and/or Dense layers, number of units for each layer…)