Weicong Feng

Professor Konstantinos Krampis

BIOL79303

10/05/2021

Assignment5

Assignment 5

Due date: Wednesday, 10/06/2021

**Each of the 3 questions below is worth 33.333 points (you get 0.001 free to reach 100).**

This homework needs to be completed on the Google Collaboratory, and the results submitted as screenshots in a .doc or .pdf. Please include the completed run of the corresponding code the question refers too along with your written answer (you can include additional code if you want). You will need to

Also you are welcome to run Tensorflow code outside of the Collaboratory, if you have such a setup, please note though that the submission need to follow the same format, meaning code cells –> output as shown on the Collaboratory (for example do not submit Python interactive command terminal code)

In preparation for the homework, you can review again the Google Collaboratory posted in the last lecture. Please watch the following videos in order to become familiar with the Collaboratory (feel free to watch any additional on Youtube):

<https://www.youtube.com/watch?v=i-HnvsehuSw>

<https://www.youtube.com/watch?v=RLYoEyIHL6A>

For the codes in the questions below, each shaded box showing a code can run in a separate Google Collaboratory cell. **Note:** You need to run cells from top to bottom (since top code cells generate dependencies for the lower cells), so you have to copy-paste and run the code cells in your own Google Collaboratory, in the same order shown in the code each question points you too. Then as the questions request you to do (for example, adjusting the number of epochs), you have to edit the code in the corresponding cells and re-run each cell. If you are still confused on how this works, re-watch the above videos with tutorials on the Google Collaboratory and also additional videos.

**Question 1.**

**NOTE:** Use instead of “from keras.layers.normalization import BatchNormalization” the “from keras.layers import BatchNormalization”.

Run the following code on the Collaboratory. Tip: If you are logged in your Google account and click the “Copy to Drive” button on the top. This will make a full copy of this Google Collaboratory sheet under your own account, and save you a lot of typing and copy-pasting compared to starting a new sheet and transferring everything over manually.

https://colab.research.google.com/github/AviatorMoser/keras-mnist-tutorial/blob/master/MNIST in Keras.ipynb

1. How many different types of neural networks (and what kind of networks) are being used to classify the digits – show the corresponding part of the code where these networks are implemented.

Ans: FCN and CNN are used to classify the digits in the notebook.

FCN:

Text

Description automatically generated

Table

Description automatically generated

CNN:

Graphical user interface, text, application

Description automatically generated

Table

Description automatically generated

1. Run the code with both types of neural networks that are in it, based on the metrics, which one does it classify the digits better? Please explain your answer by also defining the metrics (so you understand what each metric means).

Ans: The performance of CNN is better than that of FCN, because the FCN got a test accuracy of 0.9784, while the CNN obtained 0.9924. Both of them use accuracy as metric, which is the proportion of true results among the total number of cases, accuracy = (TP+TN)/(TP+TN+FP+FN).

1. Could you try a different activation function instead of softmax in the final layer and see what happens with the model predictions and its metrics? Choose one from the list <https://www.tensorflow.org/api_docs/python/tf/keras/activations>

Ans: The accuracy doesn’t change a lot when the metric is changed to sigmoid. However, the relu gave particular decrease, the accuracy of FCN is low to 0.8665, and that of CNN is 0.942. The activation function of linear lead to significant a cliff fall in performance of both FCN and CNN, the accuracies are only more than 0.1.

**Question 2.**

Run the following code on the Collaboratory (you can skip the part showing the images if you wish). You will need to copy this code in a new, clean sheet of the Google Collaboratory.

<https://www.tensorflow.org/tutorials/load_data/images>

1. Modify the number of Convolutional and Max Pooling layers, for example add a pair or two, and remove a layer or two :

model = tf.keras.Sequential([  
  tf.keras.layers.experimental.preprocessing.Rescaling(1./255),  
  tf.keras.layers.Conv2D(32, 3, activation='relu'),  
  tf.keras.layers.MaxPooling2D(),  
  tf.keras.layers.Conv2D(32, 3, activation='relu'),  
….

Then rerun the training with the modifications

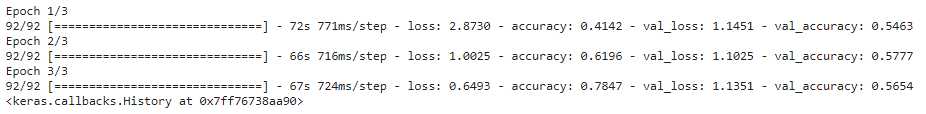
model.compile(  
 …

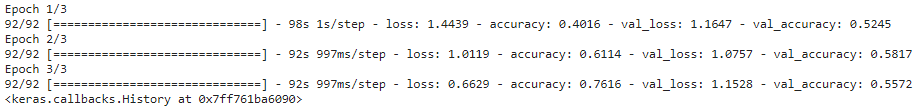
and also

model.fit(  
  train\_ds,

..

What do you observe changing in the metrics? (Just run it for 3 epochs as it is)





A picture containing diagram

Description automatically generated

Diagram

Description automatically generated with medium confidence

A picture containing diagram

Description automatically generated

Ans: Above screenshots are the training results of network with 1 to 5 hidden layers from top to bottom. As the number of hidden layers decrease, the training accuracy increases while the validation accuracy increases. Theoretically, the complex models with more layers are easy to overfit where is different from what we observe here.

1. Modify the number of epochs increasing them gradually (you might reach a point where it gets too slow in the Google Collaboratory). What do you observe in the metrics as you increase the epochs, is there a point where the metrics plateau?

Ans: After 20 iterations, the training accuracy closes to 1.0, and the validation accuracy stays at 0.62 or 0.63. After 250 iterations, the validation accuracy begins declining.

1. In which part of the code, we split the dataset in training / validations and what portions? What is the purpose of doing this?

Ans: In the “create a dataset” section, tf.keras.utils.image\_dataset\_from\_directory is used to create and split datasets. The training data set is used to train the model, and the model learn from training data set. The validation data set is used to evaluate the model performance to make sure the generalization of the model by using data set that the model hasn’t seen.

1. Look at the structure of the Convolutional Neural Network as specified in the code for this image classification example <https://www.tensorflow.org/tutorials/images/classification>. What are the differences? Make those adjustments to modify the code you just made on a – c above and re-run the model (use 5 epochs or so). What do you observe in the model metrics?

Ans: Previous model includes 3 convolution layers with same filter size while this model uses 3 convolution layers with various filter size. The new network achieves validation accuracy of 0.61 at 2nd epoch and is more robust, that keeps the validation accuracy of 0.62 until 500 epochs.

**Question 3.**

Run the following code on Deep Learning for genomics on the Google Collaboratory:

<https://colab.research.google.com/github/TankMermaid/1000-genomes-genetic-maps/blob/master/A_Primer_on_Deep_Learning_in_Genomics_Public.ipynb>

1. Describe in a couple of sentences the overall function of this neural network for bioinformatics predictions – what the predictions taking place, what are the data used, and what type of neural network we are using? From which parts of the code, you can find the answers to each of these points?

Ans: In this notebook, the labeled DNA sequences is used for input data, and the target is predicting the binding motifs in DNA or not. Following codes loads input data set and labels:

Graphical user interface, text, application, email

Description automatically generated

Graphical user interface, text, application

Description automatically generated

The 1D CNN is used for the model:

Table

Description automatically generated

1. How many prediction classes this neural network has, and describe what are these classes? In addition to finding this from the text cells in the code, please also point the parts of the actual code that would demonstrate the number of prediction classes (it should be one of the final layers in the network).

Ans: There are 2 classes in this CNN, “1” for a protein bounding to the sequence, and “0” for not. In the code, the output layer in the CNN has 2 neurons that is going to output two classes. The code is : model.add(Dense(2, activation='softmax')).

1. What portion of the data we use for training, validation and testing? Where do you see that in the code?

Ans: The whole data set is split to two parts by train\_features, test\_features, train\_labels, test\_labels = train\_test\_split( input\_features, input\_labels, test\_size=0.25, random\_state=42), training data set accounts for 75%, and test data set 25%.

While fitting, the train data set is split to training data set and validation data set by history = model.fit(train\_features, train\_labels, epochs=50, verbose=0, validation\_split=0.25).

In the evaluation, the test data set is used for evaluating the performance of the network in history = model.fit(train\_features, train\_labels, epochs=50, verbose=0, validation\_split=0.25).

1. Run the code in your Google Collaboratory up to the point where we have the model lost / accuracy graphs (including printing these graphs). What do you observe in these graphs if you modify the testing and validation portions of the datasets? You would need to re-run the cells from all the way up (where we define the training / validation portions) up and including the cells generating the graphs. Similarly, if you reduce significantly the number of epochs, what do you observe in those graphs?

Ans: original (25% for validation)

Chart, histogram

Description automatically generatedChart

Description automatically generated

Change 15% validation, don’t a lot.

Chart, histogram

Description automatically generatedChart

Description automatically generated

When the validation is changed to 45%, the convergence become slower.

Chart, line chart

Description automatically generatedChart

Description automatically generated

If the epochs are decrease to 10, the model haven’t converged, so loss higher and accuracy lower. However, you can find the validation loss and accuracy are closer that of training.

Chart, line chart

Description automatically generatedChart, line chart

Description automatically generated