Fall 2021: CSEE5590/490 – Special Topics

Python and Deep Learning Module-2 - ICP-10

Lesson Overview:
In this lesson, we are going to discuss Image classification with CNN.

Use Case Description:
Image Classification with CNN
1. Training the model
2. Evaluating the model

Programming elements:
1. About CNN
2. Hyperparameters of CNN
3. Image classification with CNN

Source Code:
Provided in your assignment folder and assignment repo.

In class programming:

1. Follow the instruction below and then report how the performance changed.(apply all at once)

Convolutional input layer, 32 feature maps with a size of 3×3 and a rectifier activation function.
Dropout layer at 20%.
Convolutional layer, 32 feature maps with a size of 3×3 and a rectifier activation function.
Max Pool layer with size 2×2.
Convolutional layer, 64 feature maps with a size of 3×3 and a rectifier activation function.
Dropout layer at 20%.
Convolutional layer, 64 feature maps with a size of 3×3 and a rectifier activation function.
Max Pool layer with size 2×2.
Convolutional layer, 128 feature maps with a size of 3×3 and a rectifier activation function.
Dropout layer at 20%.
Convolutional layer,128 feature maps with a size of 3×3 and a rectifier activation function.
Max Pool layer with size 2×2.
Flatten layer.
Dropout layer at 20%.
Fully connected layer with 1024 units and a rectifier activation function.
Dropout layer at 20%.
Fully connected layer with 512 units and a rectifier activation function.
Dropout layer at 20%.
Fully connected output layer with 10 units and a softmax activation function

2. Change the previous model into Keras Functional API model.

   2.1 Apply the following callbacks to the model:

      - ModelCheckPoint.

      - EarlyStopping.

3. Predict the first 4 images of the test data. Then, print the actual label for those 4 images (label means the probability associated with them) to check if the model predicted correctly or not.

4. Build your own dataset by collecting images from the internet for example:

- Transportation images (Airplanes, Trains, Cars, ..)
- Animals (Cats, Dogs, ..)

4.1 Train the model on your dataset and report the accuracy and type of pre-processing that needed to be done.

4.2 Plot the training and validation accuracy.

4.3 Save the model as a file and load the model again and predict on some images.

** Follow the IPC rubric guidelines.

Submission Guidelines:

1. Once finished present your work to TA during class time.
2. Once evaluated submit your source code and documentation to GitHub and represent the work in a ReadMe file properly (short summary for the ICP).

After class submission:

1. Complete your work and submit to your repo before the deadline.
2. Record a short video (1~3) minute, explaining the technical part and method used.
3. Add video link to ReadMe file.

Note: *Cheating, plagiarism, disruptive behavior and other forms of unacceptable conduct are subject to strong sanctions in accordance with university policy. See detailed description of university policy at the following URL: https://catalog.umkc.edu/special-notices/academic-honesty/*