Summer 2021: CSEE5590 – Special Topics

Python_Lesson_7_Lecture_1: CNNs

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. Did the performance change?

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

4. Predict the first 4 image 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

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

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

5.1 Train the model on your dataset and report the accuracy.

5.2 Plot the training and validation accuracy.

6. Apply the following callbacks:
   - ModelCheckPoint.
   - EarlyStopping.

Online Submission Guidelines (for Online students):
1. Submit your source code and documentation to GitHub and represent the work in a ReadMe file properly (submit your screenshots as well. The screenshot should have both the code and the output)
2. Comment your code appropriately
3. Video Submission (2 – 5 min video showing the demo of the assignment, with brief voice over on the code explanation)

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/*