AML&XAI

Assignment 1 : Regularization-based Continual Learning

1. Goal

Solve the continual learning problem via regularization-based method

2. Development environment

- python (>=3.6) on Linux
- pytorch (>=1.6)

3. Template file

Uncompleted code zip 1

- Execution using main.py
    - Example of execution command

      python3 main.py --trainer ewc --dataset CIFAR100 --nepochs 60 --lr 0.001 --lamb 10
- Trainer folder : There are three files including the following methods
    - Vanilla (no regularization)
    - Elastic weight consolidation (EWC)
    - L2 (Use identity matrix instead of fisher information in EWC)
- Data_handler folder : Pre-defined loader that produces sequential tasks by processing whole datasets (MNIST, CIFAR-100)
- Network folder : There exists two kinds of neural network architectures
    - MNIST dataset is trained with 2 layer MLP network
    - CIFAR-100 dataset is trained with 6 layer convolution network

4. Experiment details

- Epoch : MNIST (20), CIFAR-100 (60)
- Optimizer : ADAM
- Learning rate : 0.001

- Learning decay : Use the value shown in the arguments.py
- Batch size : 256
- Seed : 0, 1, 2, 3

5. Continual learning scenario

- Split MNIST : 5 sequential tasks (Each task consists of two classes)
- Split CIFAR-100 : 20 sequential tasks (Each task consists of five classes)

6. To Do

- Complete three script files for each method (vanilla, EWC, L2) in Trainer folder
- For the regularization strength 'lambda' on L2 and EWC, search the hyperparameter, and submit the plots for the results. The number of total figures is four (2 scenarios per method)
    - Do the experiments only on seed 0
    - You can choose the range for searching the hyperparameter, but you must have the figures include at least five hyperparameters
    - x-axis and y-axis of each plot represent the number of tasks learned so far and the average accuracy, respectively, and each plot should have average accuracy curve per hyperparameter (i.e., at least five curves per plot)
- Using the best hyperparameter (**'best' means the highest final average accuracy for test set**) selected for L2 and EWC, submit plots for each scenario to compare the performance of three methods (2 figures).
    - The results should be averaged over four random seeds.
    - x-axis and y-axis of each plot represent the number of tasks learned so far and the average accuracy, respectively, and each plot should have three lines.

7. Scoring rule

- Total 100 points
    - 10 points: Vanilla and L2 are implemented well
    - 20 points: EWC is implemented well
    - 20 points: Figures for hyperparameter search
    - 20 points: Figures for comparing three methods

- 10 points: Report (You can use any editor, e.g., word, latex etc.)
- 20 points: Base points

8. Submission due

- You should submit three source files for methods (vanilla.py, l2.py, ewc.py) and the report until 2022. 4. 11. 11:59 pm.