Final Test

[ECE30001] Deep Learning Application

공개 정보 활용 허용 범위

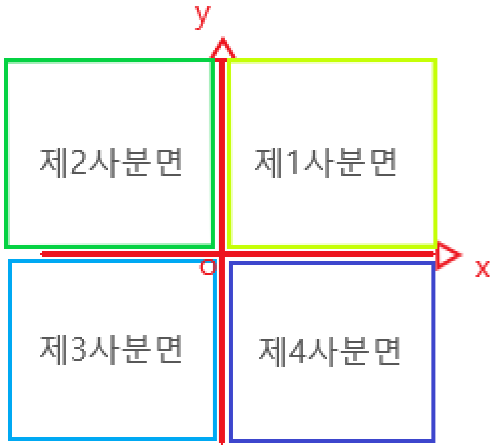
- 개인별 시험입니다. 다른 사람의 도움을 받거나 토론, 정보공유 등은 부정행위에 해당합니다.
- 본 과목에서 제공한 강의자료, 과제 및 소스코드, 본인이 작성한 노트, 본인이 작성한 과제 답안은 사용 가능합니다.
- 논문이나 책도 참조 가능합니다. 단, 소스코드가 포함된 책은 금지합니다.
- 인터넷 검색을 통해 주요 함수 및 class 사용법(예: 클래스명, 함수명 및 파라미터 등)을 확인하는 것은 허용됩니다.
- 오픈 소스나 타인이 만든 코드를 사용하는 것은 금지됩니다.
- Q&A는 정오까지 가능합니다. 오픈 톡방을 통해 질문하세요. (단, 소스 코드를 공유하면 F가 주어집니다.)

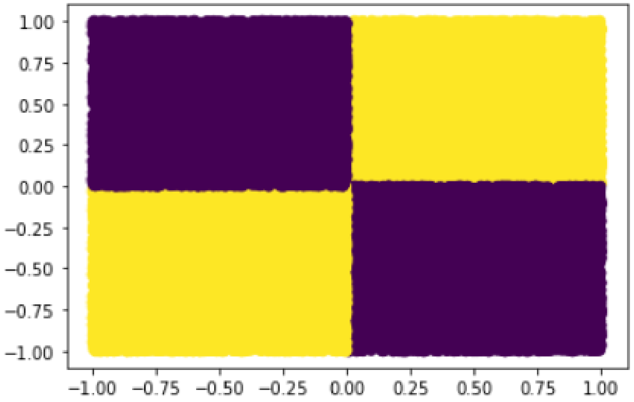
Mission

- Solve Problem 1 and 2 on Colab and submit the .ipynb files on HISNet.
  - Submit the following files on HISNet report board
    - Final1_<student_id>_<name>.ipynb # Solution of Problem #1
    - Final2_<student_id>_<name>.ipynb # Solution of Problem #2
  - Closing time: 23:50:00.
    - Submission deadline: 24:00:00
    - The 10 minutes (23:50~24:00) is for submission.
    - No submission will be accepted after 24:00:00.

Problem #1 Extended XOR Problem

- Create 'extended XOR' dataset
  - Using numpy, generate data samples whose distribution is similar to the right figure.
    - Fill each quadrant with 10,000 random samples (totally, 40,000 samples)
      - The horizontal and vertical coordinate should be in range of [−1, 1].
    - Assign label ‘class 0’ to the samples in 1st and 3rd quadrants (yellow regions)
    - Assign label ‘class 1’ to the samples in 2nd and 4th quadrants (purple regions)
  - Draw a scatter diagram similar to the right figure using matplotlib.
  - Randomly split the samples into training samples (75%) and test samples (25%).





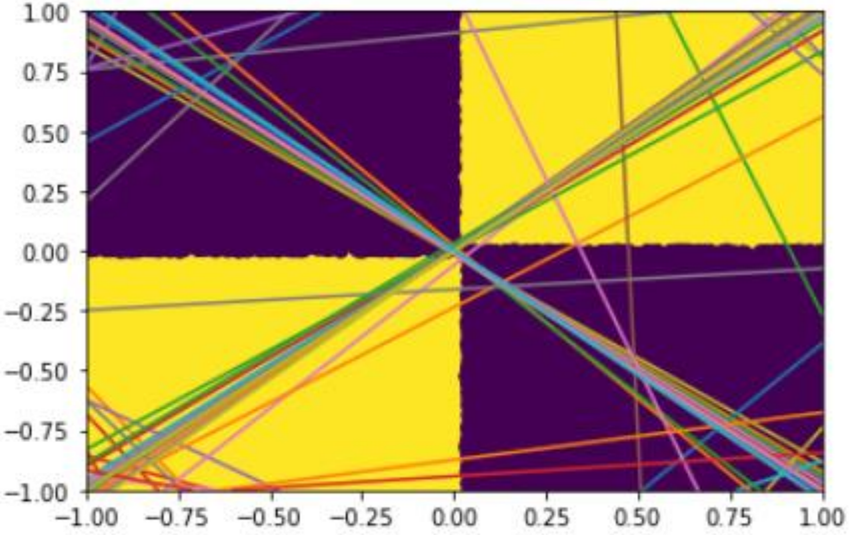
40,000 random samples from uniform distribution
Each quadrant is filled with 10,000 samples

Problem #1 Extended XOR Problem

- Build a classifier to predict the extended XOR samples using MLP using the training samples.
  - Use scikit-learn or pytorch.
  - The accuracy should be at least 99%.
  - Use few hidden nodes as possible.
    - You'll get a point inversely proportional to the number of hidden nodes.
  - Use as many hidden layers as you want.

Problem #1 Extended XOR Problem

- Visualize the 1st hidden layer of the trained MLP on the scatter diagram as the following figure.
  - Draw the lines '$ax + by + c = 0$', where a and b are the input weights a hidden node and c is its bias.
  - Set the horizontal and vertical range to [−1, 1].



< e.g. visualization of 100 hidden nodes >

Problem #2

- Build a image recognizer using PyTorch.
  - Download data.tgz from the following link.
    - https://drive.google.com/file/d/1IPJ6_LciQMBJ6YUzp-PD50Y7SPvpJi_4/view?usp=sharing
    - You need to login by an id of handing.edu.
  - Upload data.tgz on colab.
    - UNIX command to uncompress: tar xvfz data.tgz
  - Create a DataLoader for the image data.
  - Build and train a CNN model to recognizer the images
  - Try to achieve as high accuracy as possible.
    - Display the accuracy.
  - The training time should be no longer than 15 min.
    - Exceeding training time can be penalized.