Assignment 2:

Object Recognition/Classification

Computer Vision

National Taiwan University

Spring 2021

Outline

Part 1: Bag-of-Words Scene Recognition

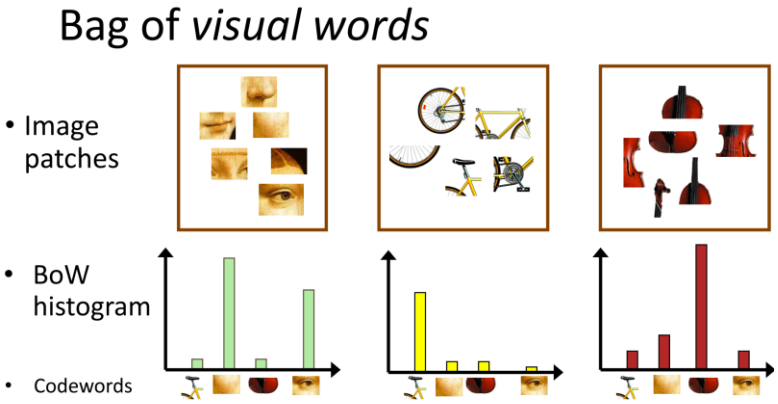
Part 2: Handwritten Digits Classification using CNN

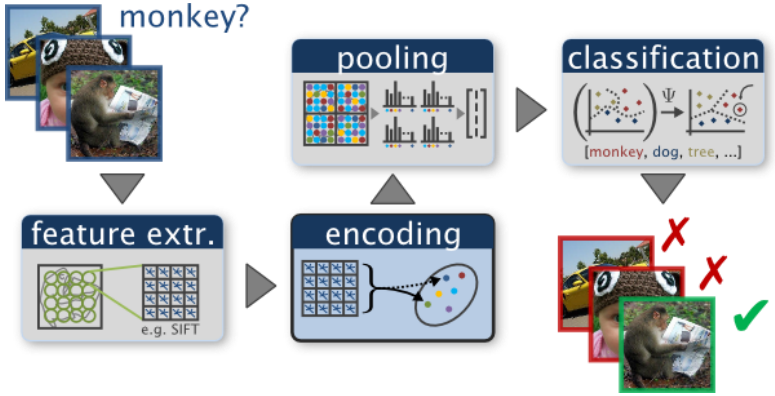
Part 1:

Bag-of-Words Scene Recognition

BoW Scene Recognition

- Tiny images representation and nearest neighbor classifier
- Bag of SIFT representation and nearest neighbor classifier





Reference: http://media.ee.ntu.edu.tw/courses/cv/21S/slides/cv2021_lec05.pdf, page 46-49

Assignment Description

- part1/p1.py
  - Read image, construct feature representations, classify features, etc.
- part1/get_tiny_images.py
  - Build tiny images features.
- part1/build_vocabulary.py
  - Sample SIFT descriptors from training images, cluster them w/ kmeans and return centroids.
- part1/get_bags_of_sifts.py
  - Construct SIFT and build a histogram indicating how many times each centroid was used.
- part1/nearest_neighbor_classify.py
  - Predict the category for each test image.
  - CAN NOT USE sklearn.neighbors.KNeighborsClassifier

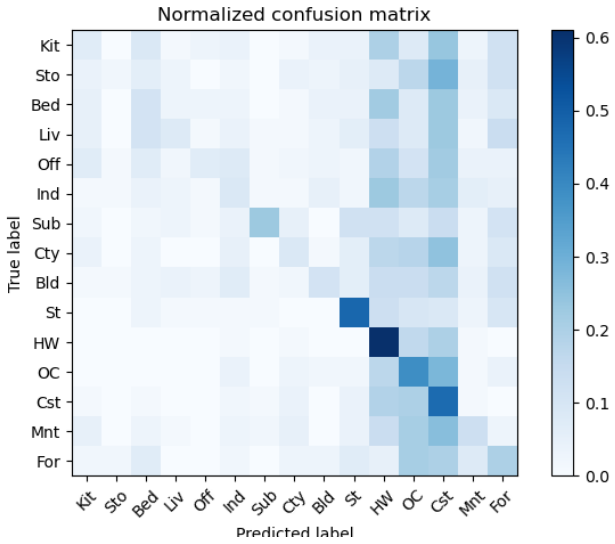
Assignment Description

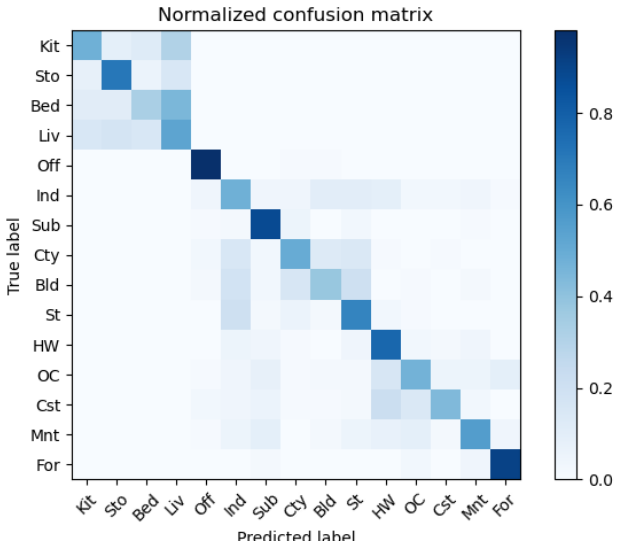
- hw2_data/p1/train
  - 100 images per category
- hw2_data/p1/test
  - 100+ images per category

```
p1
├── test
│   ├── Bedroom
│   ├── Coast
│   ├── Forest
│   ├── Highway
│   ├── Industrial
│   ├── InsideCity
│   ├── Kitchen
│   ├── LivingRoom
│   ├── Mountain
│   ├── Office
│   ├── OpenCountry
│   ├── Store
│   ├── Street
│   ├── Suburb
│   └── TallBuilding
└── train
    ├── Bedroom
    ├── Coast
    ├── Forest
    ├── Highway
    ├── Industrial
    ├── InsideCity
    ├── Kitchen
    ├── LivingRoom
    ├── Mountain
    ├── Office
    ├── OpenCountry
    ├── Store
    ├── Street
    ├── Suburb
    └── TallBuilding
```

Assignment Description

- Result visualization – confusion matrix



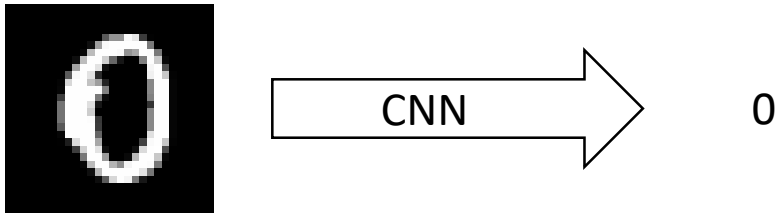


Part 2:

Handwritten Digits Classification using CNN

Digit Classification using CNN

- In this assignment, you will need to perform image classification which predicts labels to each image with CNN models.
  - Input : monochrome image
  - Output : classification label



Assignment Description

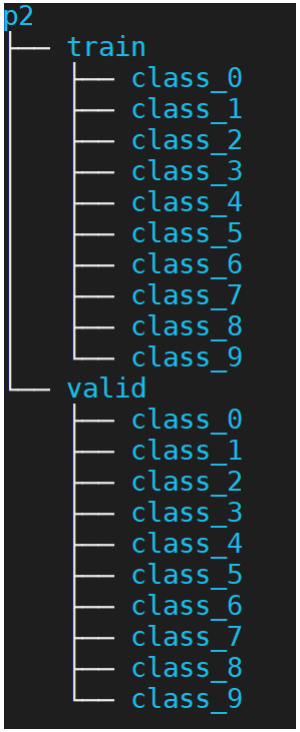
- part2/model.py
  - Build ConvNet(you may use LeNet-5 as baseline model)
  - Implement improved CNN model
  - Baseline and improved model CAN NOT import torchvision.models
- part2/data.py (DO NOT MODIFY)
  - Prepare dataset and dataloader
- part2/train.py
  - Train CNN model
- part2/eval.py
  - Evaluate model performance

```
class ConvNet(nn.Module):
    def __init__(self):
        super(ConvNet, self).__init__()
        # TODO

    def forward(self, x):
        # TODO
        return out

    def name(self):
        return "ConvNet"

class MyNet(nn.Module):
    def __init__(self):
        super(MyNet, self).__init__()
        # TODO

    def forward(self, x):
        # TODO
        return out

    def name(self):
        return "MyNet"
```

Yann LeCun, et al. “Gradient-Based Learning Applied to Document Recognition”

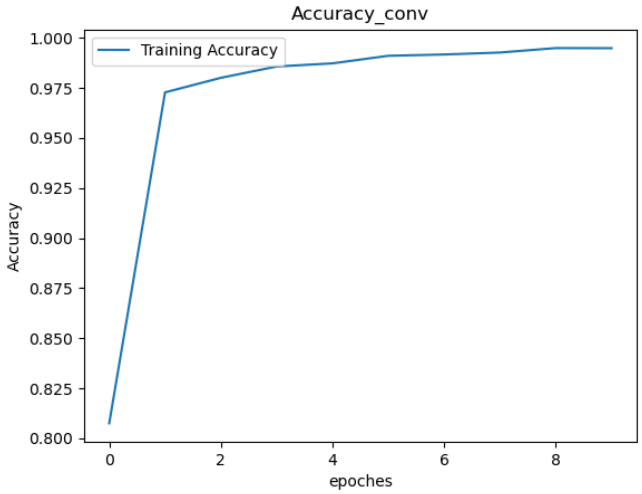
Assignment Description

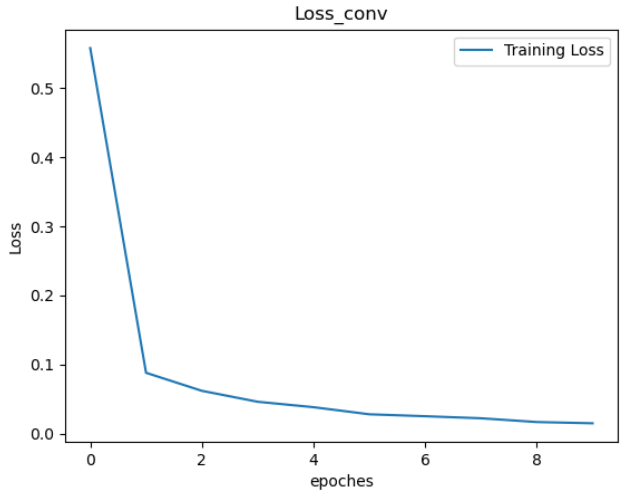
- hw2_data/p2/train, hw2_data/p2/valid
- Images collected from MNIST
- Image shape: 28 x 28 x 1 (h x w x c)
- Training images:
  - 4000 images for each class, 40000 images in total
  - images: 0000.png , … , 3999.png
- Validation images:
  - 1000 images for each class, 10000 images in total
  - images: 4000.png , … , 4999.png



Assignment Description

```
ConvNet(
  (conv1): Conv2d(1, 6, kernel_size=(5, 5), stride=(1, 1))
  (conv2): Conv2d(6, 16, kernel_size=(5, 5), stride=(1, 1))
  (fc1): Linear(in_features=256, out_features=120, bias=True)
  (fc2): Linear(in_features=120, out_features=84, bias=True)
  (fc3): Linear(in_features=84, out_features=10, bias=True)
)
```





Grading (Total 15%)

- Part 1 : 40%
  - Model Performance(25%)
    - Accuracy should be above the baseline score to get points
      - Tiny images + KNN(10%): 0.2
      - Bag of SIFT + KNN(15%): 0.6
  - TAs will execute your code to check if you pass the baseline.
  - Report(15%)
    - (5%) Report accuracy of two settings
    - (5%) Plot confusion matrix of two settings
    - (5%) Compare the results of both settings and explain the result.

Grading (Total 15%)

- Part 2 : 60%
  - Model Performance(40%)
    - Accuracy should be above the baseline score to get points
      - On the validation set(15%+15%): 0.98
      - On the test set(5%+5%): 0.96
    - Accuracy of your improved model should be higher than the baseline model
  - TAs will execute your code to check if you pass the baseline.
  - Only TAs have the test data
  - Report(20%)
    - (4%) Print the network architectures & number of parameters of both models
    - (10%) Plot the learning curve (loss, accuracy) of the training process(train/validation). Total 4 plots.
    - (6%) Compare the results of both model and explain the result.

Execution of hw2

- TAs will run your code in the following manner:
- part1:
  - python3 part1/p1.py $1 $2 $ 3
    - $1: type of feature representation (tiny images/SIFT)
    - $2: type of classifier(nearest neighbor)
    - $3: dataset path
    - E.g., python3 p1.py -- feature tiny_images – classifier nearest_neighbor – dataset_path ./data/
- part2:
  - python3 part2/eval.py $1 $2 $3
    - $1: directory of the testing images folder
    - $2: type of model (conv/mynet)
    - $3: path folder of the output prediction file
    - E.g., python3 test.py ./test/ mynet ./output/out.csv

Penalty

- If we can not reproduce your accuracy on the validation set, you will get 0 points in model performance and you will receive a 30% penalty in your report score.
- If we can not execute your code, we will give you a chance to make minor modifications to your code. After you modify your code,
  - If we can execute your code and reproduce your results on the validation set, you will still receive a 30% penalty in your homework score.
  - If we can run your code but cannot reproduce your accuracy on the validation set, you will get 0 points in model performance and you will receive a 30% penalty in your report score.
  - If we still cannot execute your code, you will get 0 in this problem.

Submission

- R07654321/
  - part1/
    - p1.py, get_tiny_images.py, build_vocabulary.py, get_bags_of_sifts.py, nearest_neighbor_classify.py
    - vocab.pkl, train_image_feats.pkl, test_image_feats.pkl
  - part2/
    - model.py, eval.py, train.py
    - Your model files
  - report_R07654321.pdf
- Compress all above files in a zip file named StudentID.zip
  - e.g. R07654321.zip
  - After TAs run "unzip R07654321.zip", it should generate one directory named "R07654321"
  - Don't upload your dataset.
  - If any of the file format is wrong, you will get zero point.

Submission

- Submit to NTU COOL
- Deadline: 4/29 11:59 pm
  - Late policy: http://media.ee.ntu.edu.tw/courses/cv/21S/hw/cv2021_delay_policy.pdf
- Do not delete your trained model before the TAs disclose your homework score and before you make sure that your score is correct.
- Note that you should NOT hard code any path in your file or script
- Your testing code have to be finished in 10 mins.
- We will execute you code on Linux system, so try to make sure your code can be executed on Linux system before submitting your homework.

Packages

- python: 3.6
- numpy: 1.19.2
- cyvlfeat: 0.5.1
- matplotlib: 3.3.4
- pillow: 8.1.2
- scipy: 1.5.2
- opencv-python: 4.5.1
- sklearn: 0.24.1
- pytorch: 1.5.1 (https://pytorch.org/get-started/locally/)
- torchvision: 0.6.1
- And other standard python packages
- E-mail or ask TA first if you want to import other packages.

TA information

- Bo-Ying Chen (陳柏穎)
  E-mail: byc@media.ee.ntu.edu.tw
  TA time: Thu. 14:00 - 15:30
  Location: 博理 421

- Chih-Ting Liu (劉致廷)
  E-mail: jackieliu@media.ee.ntu.edu.tw