Artificial Intelligence

Assignment 1

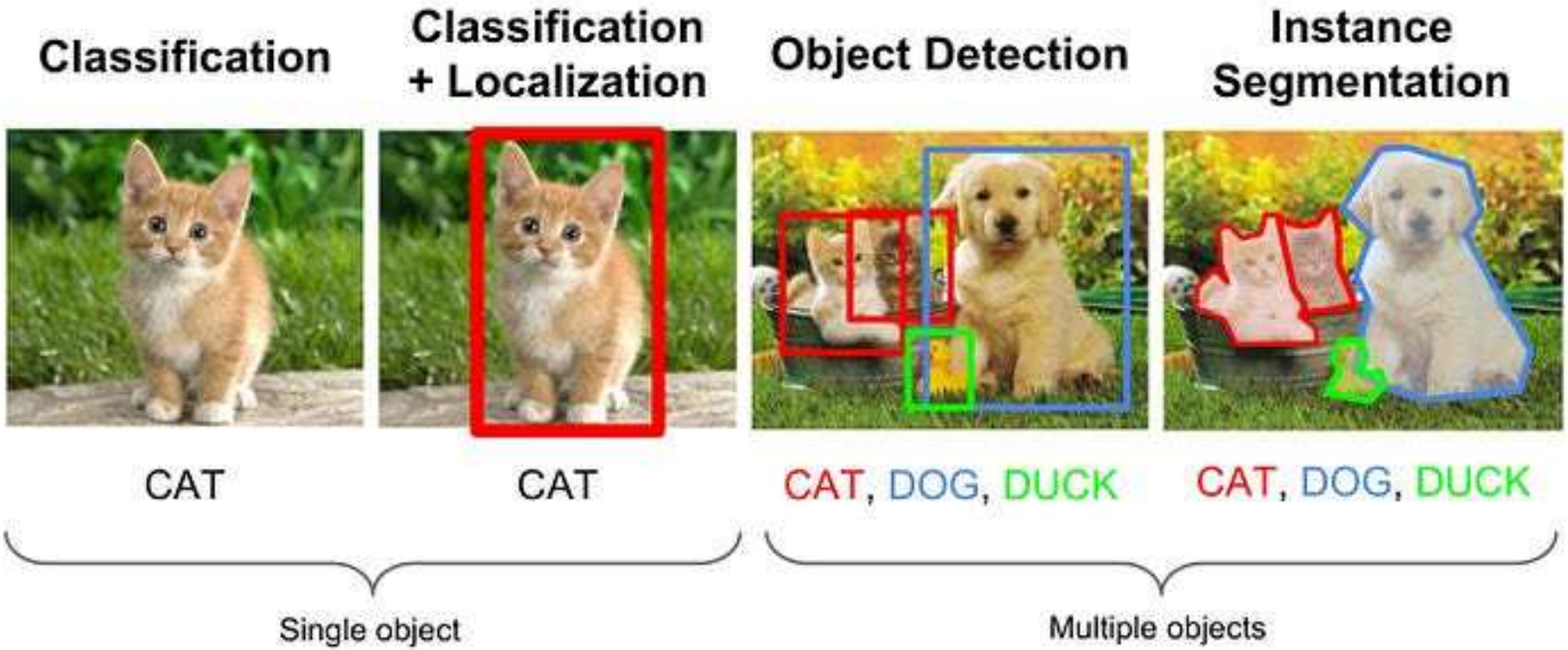
Dahuin Jung

School of Computer Science and Engineering

Soongsil University

2024

Computer vision tasks



https://medium.com/comet-app/review-of-deep-learning-algorithms-for-object-detection-c1f3d437b852

- Besides, image reconstruction, image synthesis, style transfer, etc…

Assignment_1-1 Objective (Convolutional neural network)

- Problem 1: Simple CNN model training on CIFAR-10 dataset
  - Implement the model as presented in the problem

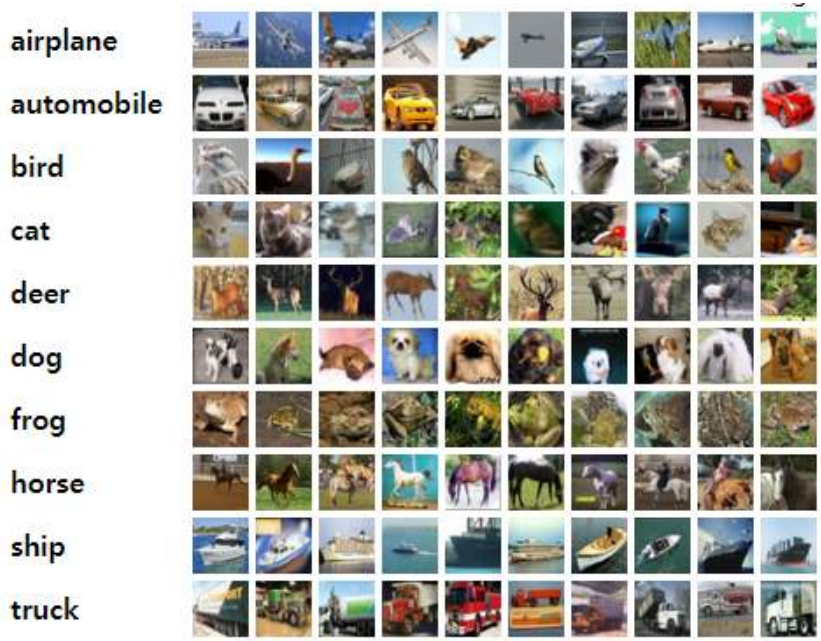
InceptionNet, ResNet 모듈 구현

- Problem 2: Inception module & Residual block implementation
  - Implement the model as presented in the problem
  - Hyperparameters can be modified (e.g., number of filters)

본인만의 독자적인 CNN Architecture 구성 과제

- Problem 3: CNN Model Training Using Problem 2 (CIFAR-10)
  - Test set accuracy ≥ 70%
  - Description of the implemented model

CIFAR-10 dataset

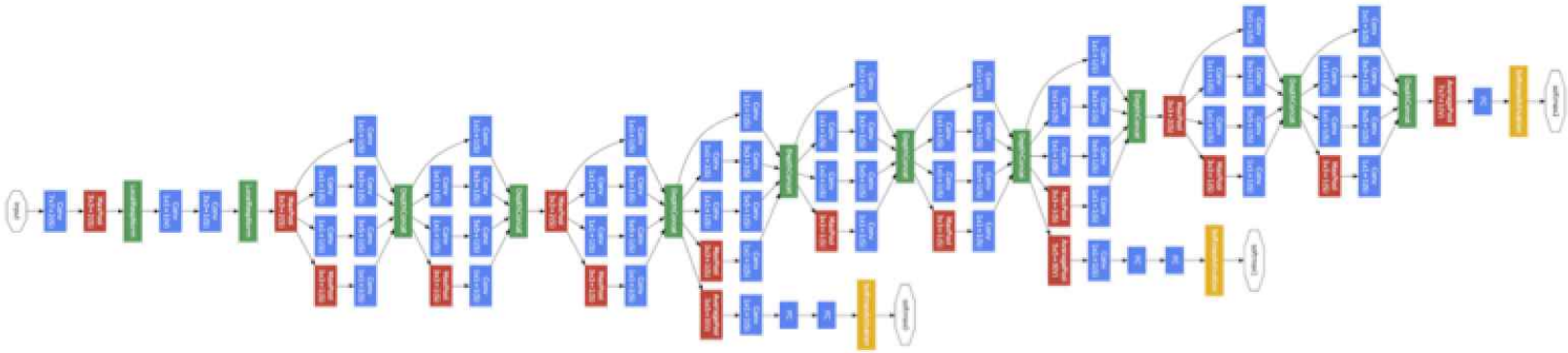


Source: https://www.cs.toronto.edu/~kriz/cifar.html

- Collection of images used to train machine learning and computer vision algorithms
- Consists of 60,000 32x32 color images in 10 classes, with 6,000 images per class
- There are 50,000 training images and 10,000 test images.
- The classes are completely mutually exclusive (no overlap between truck & automobile).

두 클래스 사이 중복되는 정보가 없다.
= automobile 클래스에는 트럭이 없음.

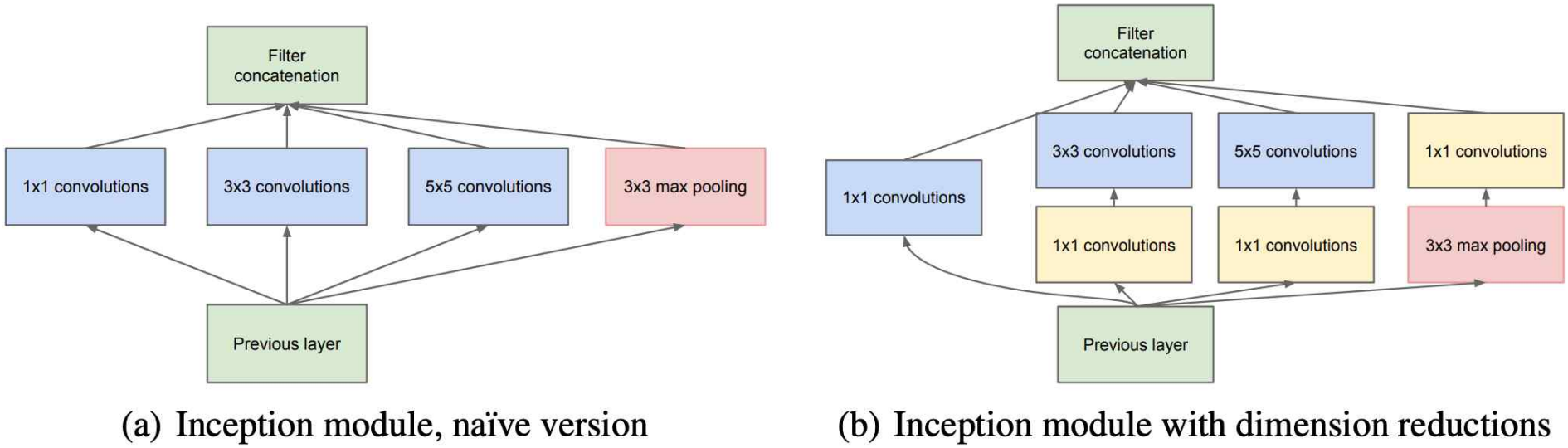
CIFAR-10 dataset



Source: Going Deeper with Convolutions (CVPR 2015)

- Deeper network with computational efficiency
  - ImageNet Large Scale Visual Recognition Competition (ILSVRC) 2014 winner
  - 22 layers with 5 million parameters (12x less than AlexNet *ILSVRC 2012 winner)
  - Efficient "Inception" module

Inception module

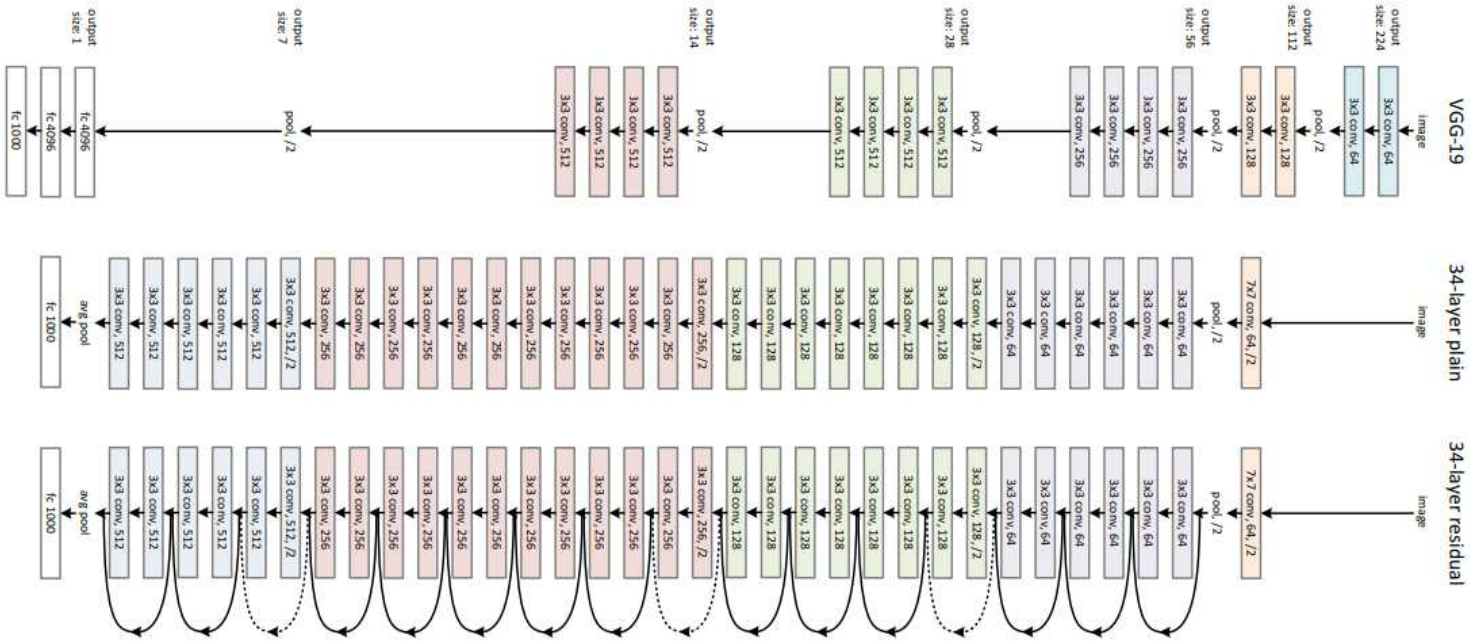


(a) Inception module, naïve version (b) Inception module with dimension reductions

Source: Going Deeper with Convolutions (CVPR 2015)

- Local network topology composing the Inception model
  - Apply parallel filter operations on the input from previous layer
  - Multiple filter sizes for convolution (1x1, 3x3, 5x5)
  - 1x1 convolution for dimensionality reduction

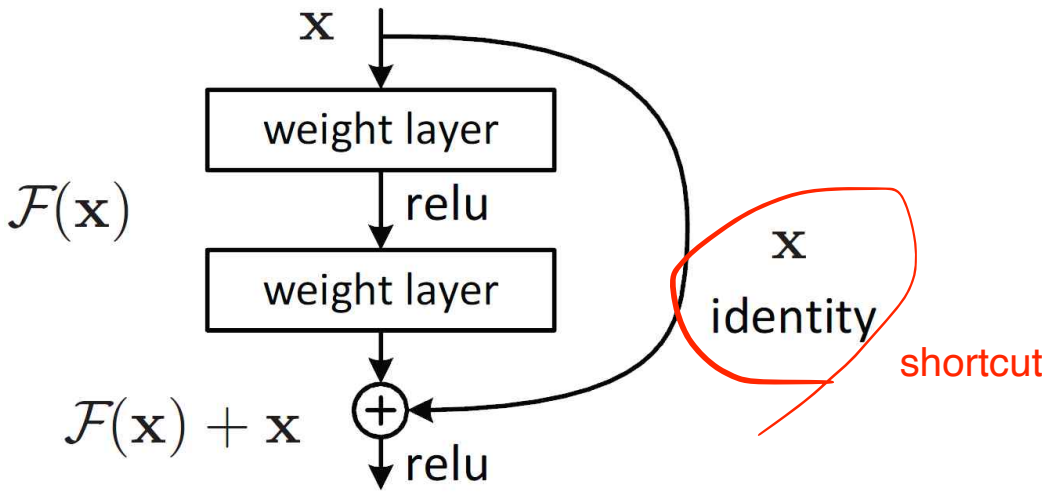
Inception module



Source: Deep Residual Learning for Image Recognition (CVPR 2016)

- Deeper network with Residual connections
  - ImageNet Large Scale Visual Recognition Competition (ILSVRC) 2015 winner
  - In a ResNet, each layer's input is added to its output, creating a "shortcut connection" or a "skip connection."

Residual Block

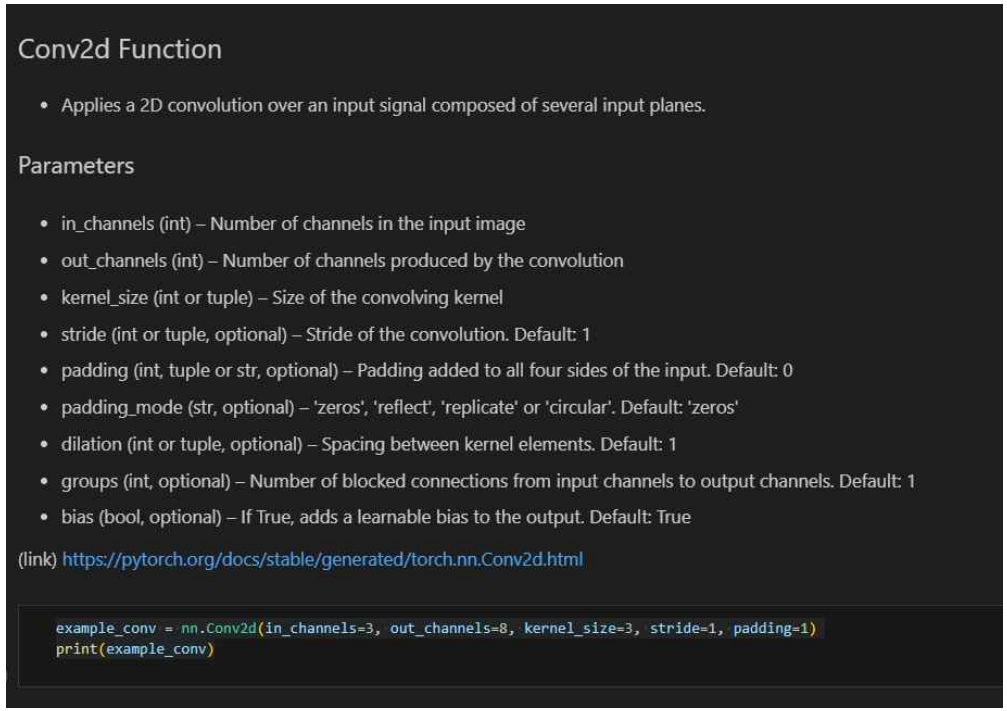


Source: Deep Residual Learning for Image Recognition (CVPR 2016)

- Deeper network with Residual connections
  - Residual connection addresses the vanishing gradient problem by introducing shortcut connections
  - The original input is added to the output of the convolutional layers.

Pytorch Function Examples

- Before writing example code, for those who find using PyTorch difficult, please make sure to refer to the following content when writing your code.(hint!)



Conv2d Function

- Applies a 2D convolution over an input signal composed of several input planes.

Parameters

- in_channels (int) – Number of channels in the input image
- out_channels (int) – Number of channels produced by the convolution
- kernel_size (int or tuple) – Size of the convolving kernel
- stride (int or tuple, optional) – Stride of the convolution. Default: 1
- padding (int, tuple or str, optional) – Padding added to all four sides of the input. Default: 0
- padding_mode (str, optional) – 'zeros', 'reflect', 'replicate' or 'circular'. Default: 'zeros'
- dilation (int or tuple, optional) – Spacing between kernel elements. Default: 1
- groups (int, optional) – Number of blocked connections from input channels to output channels. Default: 1
- bias (bool, optional) – If True, adds a learnable bias to the output. Default: True

(link) https://pytorch.org/docs/stable/generated/torch.nn.Conv2d.html

```
example_conv = nn.Conv2d(in_channels=3, out_channels=8, kernel_size=3, stride=1, padding=1)
print(example_conv)
```

Output Examples

- Assignment_1-1
- Problem 1 : Simple CNN model training on CIFAR-10 dataset

```
[ ] # load trained model then test
    net.load_state_dict(torch.load(PATH))
    print_accuracy(net, dataloader_test)
```

```
<ipython-input-9-17bdbaab9414>:2: FutureWarning: You are using `torch.load` with `weights_only=False` (the curr
  net.load_state_dict(torch.load(PATH))
Accuracy of the network on the 10000 test images: 50 %
```

- Problem 3: CNN Model Training Using Problem 2 (CIFAR-10)

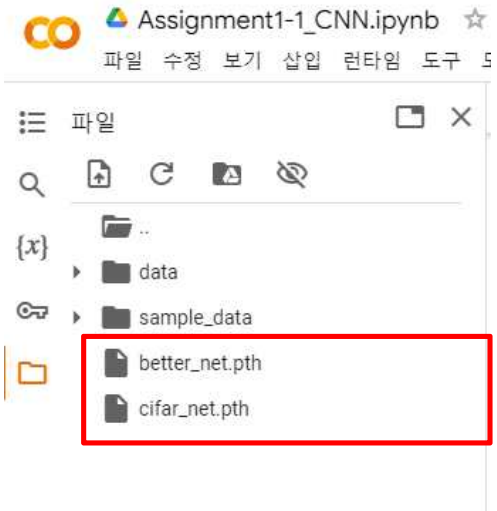
```
[8,  4000] loss: 0.161
[8,  6000] loss: 0.182
[9,  2000] loss: 0.101
[9,  4000] loss: 0.108
[9,  6000] loss: 0.125
[10,  2000] loss: 0.066
[10,  4000] loss: 0.066
[10,  6000] loss: 0.087
Finished Training
Saved Trained Model
<ipython-input-13-dcfb768c14cd>:14: FutureWarning: You a
  betternet.load_state_dict(torch.load(PATH))
Accuracy of the network on the 10000 test images: 71 %
```

How to install assignment files

- Files included: 7 in total
  - Assignment1-1_CNN.ipynb
  - CollectSubmission.sh
  - imgs and data folders (data folder is empty)

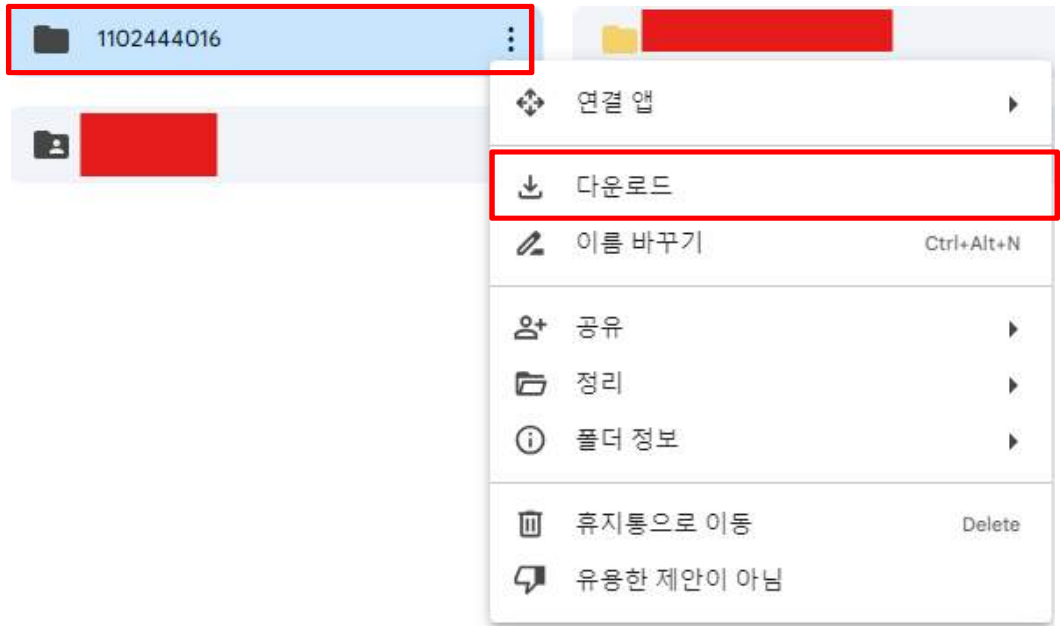
Submitting your work(Colab)

- How to save a .pth file in Colab
  - If the model trains successfully, first go to the respective part and check the .pth file.
  - Secondly, as shown in the image, select the corresponding .pth file and download it.
  - Finally, place the downloaded .pth file into the folder that was previously mentioned.



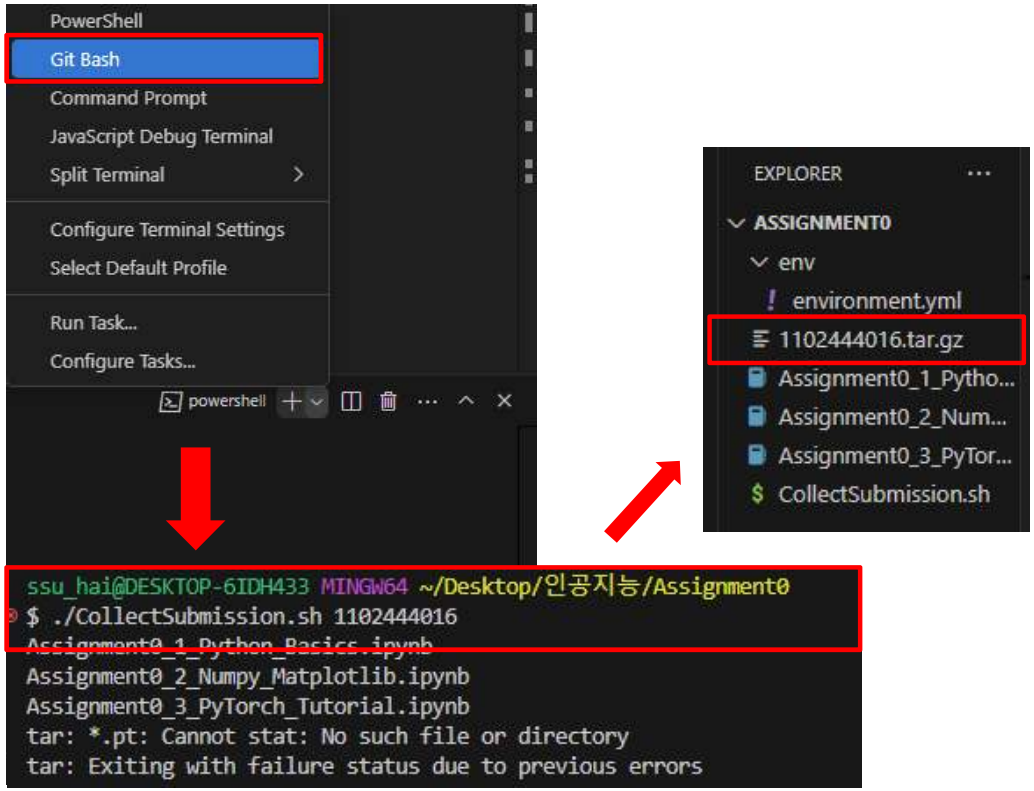
Submitting your work(Colab)

- Go into the Google Drive folder you created earlier
  - Files to be placed inside the folder: 1 ipynb files, 2 pth files
  - Folder name : student-id
  - Please send the downloaded zip file.
  - DO NOT clear the final outputs



Submitting your work(local)

- Submitting your work
  .pth(트레이닝된 모델 저장파일)도 포함해서 tar.gz로 변환됨. pth 파일 확인 필요
  - After you are done
    - Open Git Bash in VSCode or download it locally.
    - $ ./CollectSubmission.sh 1102444016 (student-id)
    - Upload the (student-id).tar.gz on ETL
  - DO NOT clear the final outputs



FAQ_1

- Q: Can I modify the batch size?
  - A: Yes, all hyperparameters can be modified.
- Q: What are n3xn3_blue and n5xn5_blue?
  - A: n3xn3_blue and n5xn5_blue refer to the number of output channels of the conv_layer. You can adjust the number of output channels as you prefer in your implementation.
- Q: When concatenating within the provided inception module, do I need to consider the tensor size?
  - A: If you check the return value of the forward method in the provided module, it uses torch.cat to concatenate y1, y2, y3, and y4 along dimension 1. In PyTorch, tensors are defined in the order of batch x channel x height x width. Therefore, the code concatenates y1~4 along the channel dimension. You only need to ensure that the sizes of batch, height, and width match, excluding the channel dimension.

FAQ_2

- Q: What is a pth file?
  - A: It is the model file saved after training. It will be automatically generated upon completion of training.
- Q: Can I also change the max iteration count, batch size of the data loader, and the number of workers?
  - A: Yes, you can. However, batch size and the number of workers mainly affect the time required for experiments and have minimal impact on performance.
  - Additionally, you can increase the number of epochs beyond the default of 2.
- Does the part where you explain the code implementation also count towards the grade?
  - A: The section explaining better_net at the end of the ipynb file is for verification purposes and does not count towards the grade.

- 질문: 코드 구현에 대한 설명 부분도 성적에 포함되나요?
- 답변: ipynb 파일의 끝에 있는 better_net에 대한 설명 섹션은 검증 목적을 위한 것이며 성적에 포함되지 않습니다.

- Q: What does the exclamation mark (!) at the beginning of the code mean?
  - A: Code following the exclamation mark (!) is shell script. It is used to run commands in the terminal.


Thank You!