NYCU DL
Lab2 - Butterfly & Moth Classification

2024 Spring

王芷鈴

Mar. 26, 2024

Lab Objective

- In this lab, you will need to analyze Butterflies or moths with one hundred species, following these three steps.

Step 1. Write your own custom DataLoader through PyTorch framework and design your own data preprocessing method.

Step 2. Classify Butterflies or moths via the VGG19 and ResNet50 .

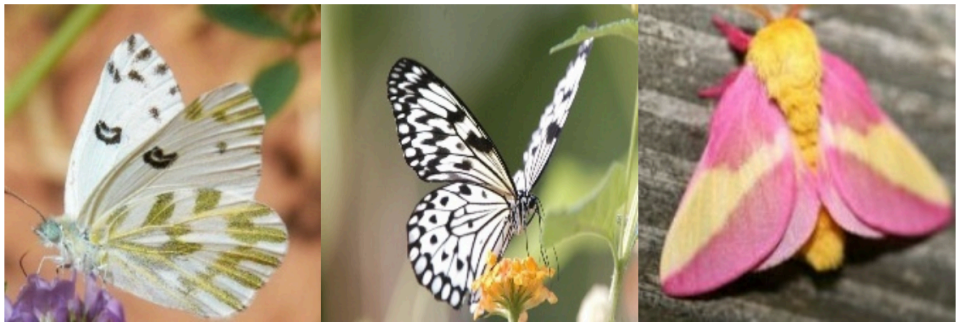
Step 3. Plot the accuracy (not loss) curve for each epoch and show the highest accuracy of two architectures.

Requirements

- Implement the **VGG19, ResNet50** architecture on your own , **do not call the model from any library**.
- Train your model from scratch, **do not load parameters** from any pretrained model.
- Implement your own custom **DataLoader**
- Design **your own data preprocessing method**
- Compare and **visualize** the accuracy trend between the **two architectures**, you need to plot each epoch **accuracy (not loss)** during training phase and testing phase.
- In the experiment results, you have to show the **highest accuracy** (not loss) of two architectures.

Dataset - Butterfly & Moths Classification

- This dataset comprises train and test datasets designed for the classification of 100 species of butterflies or moths.
- Format: 224 x 224 x 3 JPG



Training - 12,594 images

Testing - 500 images

Reference: https://www.kaggle.com/datasets/gpiosenka/butterfly-images40-species/data

DataLoader

- Implement your own custom DataLoader
- Below is the skeleton that you have to fill to have a custom dataset, refer to “dataloader.py”

```python
class BufferflyMothLoader(data.Dataset):
    def __init__(self, root, mode):
        """…
        self.root = root
        self.img_name, self.label = getData(mode)
        self.mode = mode
        print("> Found %d images..." % (len(self.img_name)))

    def __len__(self):
        """'return the size of dataset"""
        return len(self.img_name)

    def __getitem__(self, index):
        """something you should implement here"""

        """…

        return img, label
```

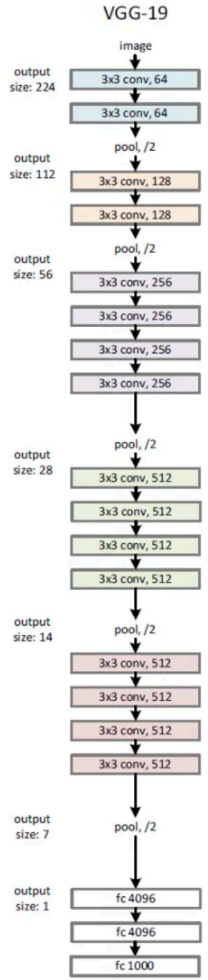
```python
def getData(mode):
    if mode == 'train':
        df = pd.read_csv('Path to train.csv')
        path = df['filepaths'].tolist()
        label = df['label_id'].tolist()
        return path, label

    else:
        df = pd.read_csv('Path to test.csv')
        path = df['filepaths'].tolist()
        label = df['label_id'].tolist()
        return path, label
```

VGGNet

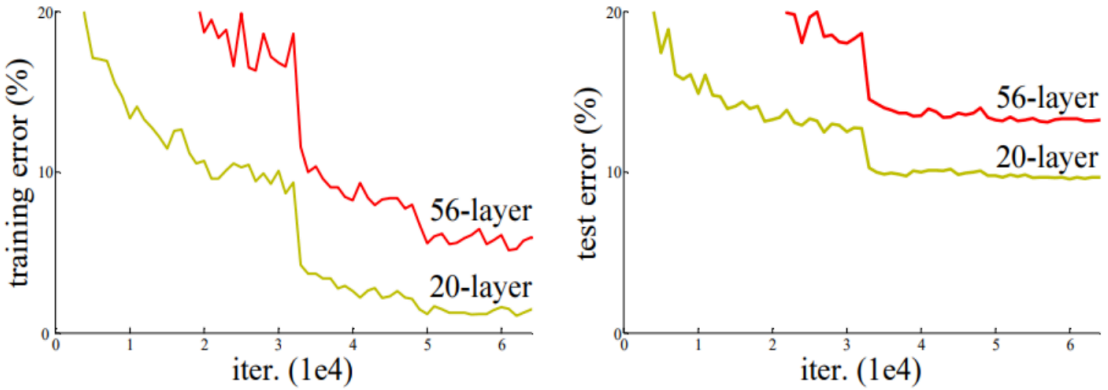
- VGGNet (Visual Geometry Group Network) is the second place of ILSVRC 2014 .
- VGG's convolutional layers leverage a minimal receptive field, i.e., 3×3,

ConvNet Configuration					
A	A-LRN	B	C	D	E
11 weight layers	11 weight layers	13 weight layers	16 weight layers	16 weight layers	19 weight layers
input (224 × 224 RGB image)					
conv3-64	conv3-64 **LRN**	conv3-64 **conv3-64**	conv3-64 conv3-64	conv3-64 conv3-64	conv3-64 conv3-64
maxpool					
conv3-128	conv3-128	conv3-128 **conv3-128**	conv3-128 conv3-128	conv3-128 conv3-128	conv3-128 conv3-128
maxpool					
conv3-256 conv3-256	conv3-256 conv3-256	conv3-256 conv3-256	conv3-256 conv3-256 **conv1-256**	conv3-256 conv3-256 **conv3-256**	conv3-256 conv3-256 conv3-256 **conv3-256**
maxpool					
conv3-512 conv3-512	conv3-512 conv3-512	conv3-512 conv3-512	conv3-512 conv3-512 **conv1-512**	conv3-512 conv3-512 **conv3-512**	conv3-512 conv3-512 conv3-512 **conv3-512**
maxpool					
conv3-512 conv3-512	conv3-512 conv3-512	conv3-512 conv3-512	conv3-512 conv3-512 **conv1-512**	conv3-512 conv3-512 **conv3-512**	conv3-512 conv3-512 conv3-512 **conv3-512**
maxpool					
FC-4096					
FC-4096					
FC-1000					
soft-max					



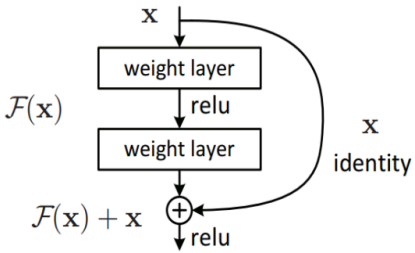
ResNet

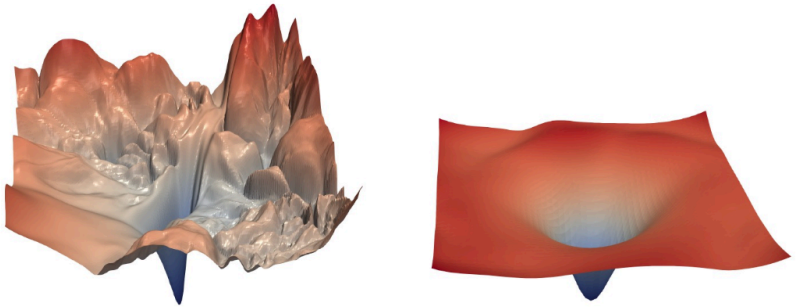
- ResNet (Residual Network) is the Winner of ILSVRC 2015 in image classification, detection, and localization, as well as Winner of MS COCO 2015 detection, and segmentation



ResNet

- To solve the problem of vanishing/exploding gradients, a skip / shortcut connection is added to add the input x to the output after few weight layers as below





(a) without skip connections (b) with skip connections

Figure 1: The loss surfaces of ResNet-56 with/without skip connections. The proposed filter normalization scheme is used to enable comparisons of sharpness/flatness between the two figures.

Source: Li, Hao, et al. "Visualizing the loss landscape of neural nets." Advances in Neural Information Processing Systems. 2018.

ResNet

- ResNet can avoid vanishing gradient problem

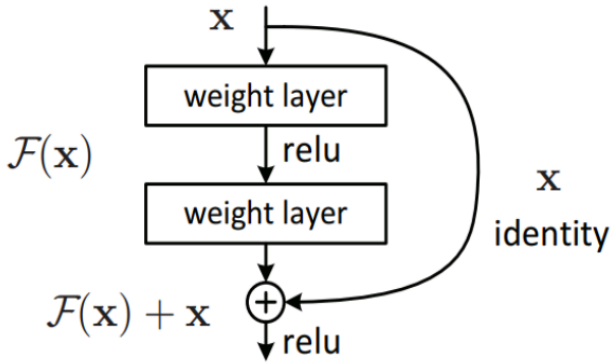
$$x \rightarrow w_1 \rightarrow \underset{y_1}{} w_2 \rightarrow \underset{y_2}{} w_3 \rightarrow \underset{y_3}{} w_4 \rightarrow \underset{y_4}{} Loss$$

$$\frac{\partial Loss}{\partial w_1} = \frac{\partial Loss}{\partial y_4}\frac{\partial y_4}{\partial z_4}\frac{\partial z_4}{\partial y_3}\frac{\partial y_3}{\partial z_3}\frac{\partial z_3}{\partial y_2}\frac{\partial y_2}{\partial z_2}\frac{\partial z_2}{\partial y_1}\frac{\partial y_1}{\partial z_1}\frac{\partial z_1}{\partial w_1}$$

$$= \frac{\partial Loss}{\partial y_4}\sigma'(z_4)w_4\sigma'(z_3)w_3\sigma'(z_2)w_2\sigma'(z_1)x_1$$

ResNet

- ResNet can avoid vanishing gradient problem

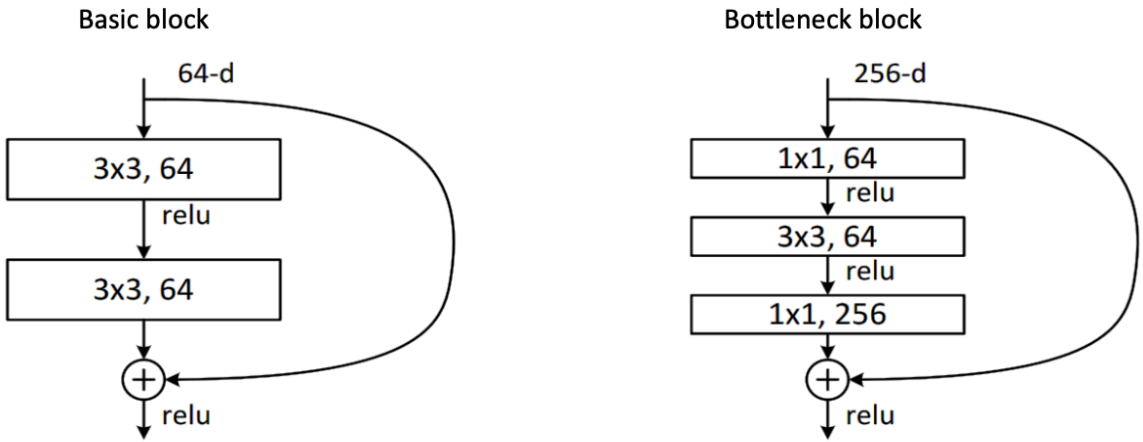


$$\mathbf{x}_L = \mathbf{x}_l + \sum_{i=l}^{L-1} \mathcal{F}(\mathbf{x}_i, \mathcal{W}_i),$$

$$\frac{\partial \mathcal{E}}{\partial \mathbf{x}_l} = \frac{\partial \mathcal{E}}{\partial \mathbf{x}_L}\frac{\partial \mathbf{x}_L}{\partial \mathbf{x}_l} = \frac{\partial \mathcal{E}}{\partial \mathbf{x}_L}\left(1 + \frac{\partial}{\partial \mathbf{x}_l}\sum_{i=l}^{L-1}\mathcal{F}(\mathbf{x}_i, \mathcal{W}_i)\right).$$

ResNet

- ResNet50 (Bottleneck block)



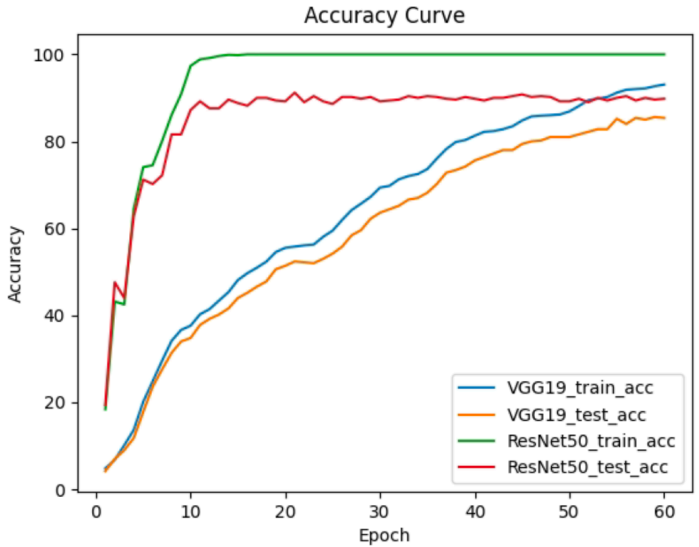
Result Comparison

- You have to show the highest accuracy (not loss) of two architectures.

```
> Found train 12594 images...
> Found test 500 images...
----------VGG19----------
VGG19           |   Train accuracy:  95.17%|   Test accuracy: 87.80%
----------ResNet50----------
Resnet50        |   Train accuracy: 100.00%|   Test accuracy: 90.00%
```

Result Comparison

- Compare and visualize the accuracy trend **between the 2 model architectures**, you need to plot each epoch accuracy (not loss) during training phase and testing phase.
- In this part, you can use the matplotlib library to draw the graph.



Report Spec (40%)

1. Introduction (10%)
2. Implementation Details (30%)
    A. The details of your model (VGG19,ResNet50)
    B. The details of your Dataloader
3. Data Preprocessing(20%)
    A. How you preprocessed your data?
    B. What makes your method special?
4. Experimental results (10%)
    A. The highest testing accuracy
        • Screenshot
        • Anything you want to present
    B. Comparison figures
        • Plotting the comparison figures
        • **(VGG19, ResNet50)**
5. Discussion(30%)
    A. Anything you want to share

Demo (60%)

---- experimental result (20%) ----

Accuracy > = 88% = 100 pts

Accuracy 85~88% = 90 pts

Accuracy 80~85% = 80 pts

Accuracy 75~80% = 70 pts

Accuracy < 75% = 60 pts

---- question (40%) ----

Score:
60% demo score (experimental results & questions) + 40% report
If the zip file name or the report spec have format error, you will be punished (-5)

Important Date

Important Date:

1. Experiment Report Submission Deadline: 4/10 (Wed) 11:59 p.m.
2. Demo date: 4/11 (Thu)

Turn in:

a. Experiment Report (.pdf)
b. Source code

Notice : zip all files in one file and name it like「DL_LAB2_YourStudentID_ name.zip」, ex: [DL_LAB2_312553037_王芷鈴.zip」

LAB timetable

	LAB1 Back-Propagation	LAB2 CNN	LAB3 CNN	LAB4 RNN+VAE	LAB5 MaskGIT	LAB6 Generative Models
Announce	3/12 (Tabc)	3/26 (Tabc)	4/2 (Tabc)	4/11 (Rn56)	5/7 (Tabc)	5/21 (Tabc)
DEMO	3/26 (Tabc)	4/11 (Rn56)	4/11 (Rn56)	5/7 (Tabc)	TBD	No demo

Reference

1. He, Kaiming, et al. "Deep residual learning for image recognition." Proceedings of the IEEE conference on computer vision and pattern recognition. 2016.

2. VGGNet - https://arxiv.org/abs/1409.1556

3. Review: ResNet - https://towardsdatascience.com/review-resnet-winner-of-ilsvrc-2015-image-classification-localization-detection-e39402bfa5d8