Lab 3

Leukemia Classification

TA 劉子齊 Jonathan

Deep Learning

NYCU CS, 2023 Summer

2023.07.27

Important Rules

Important Date :

- Closure of the Kaggle competition: 8/3 (Thu.) 11:59 a.m.
- Model weight upload deadline: 8/3 (Thu.) 11:59 a.m.
- Experiment Report Submission Deadline: 8/3 (Thu.) 11:59 a.m.
- Demo date: 8/3 (Thu.)

Model Weight Upload Form:

ResNet18 & ResNet50: https://forms.gle/F1pQn9osaZ5JzW5Y8
ResNet152: https://forms.gle/6gCvxuvFdA12Yzpz9

Files to turn in to the E3 platform: Experiment Report (.pdf) & Source code (.py)

Notice: zip all files in one file and name it like「 DLP_LAB3_your studentID_name.zip 」

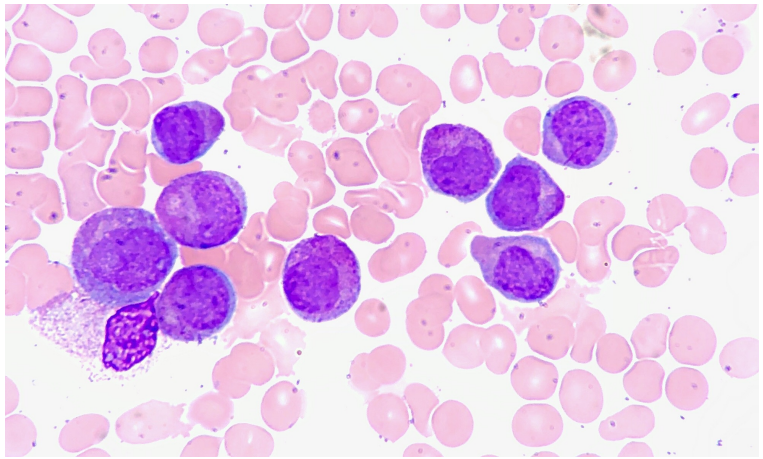
Lab Objective

- In this lab, you will need to analysis acute lymphoblastic leukemia (急性淋巴性白血病) in the following three steps.

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

Step 2. Classify acute lymphoblastic leukemia via the ResNet [1].

Step 3. Upload your prediction result to the Kaggle competition, plot the confusion matrix and the accuracy curve to evaluate the performance

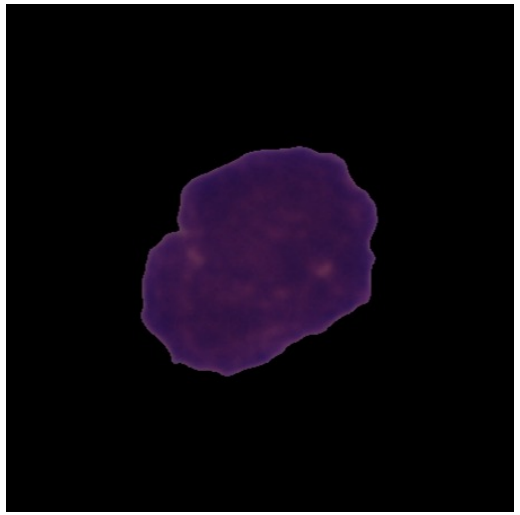


Requirements

- Implement the **ResNet18, ResNet50, ResNet152** 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.
- Compare and **visualize** the accuracy trend between the **3 architectures**, you need to plot each epoch **accuracy (not loss)** during training phase and testing phase.
- Implement your own custom **DataLoader**
- Design **your own data preprocessing method**
- Calculate the **confusion matrix** and plotting

Dataset – Leukemia Classification (Kaggle)

- Acute lymphoblastic leukemia (ALL) is the most common type of childhood cancer and accounts for approximately 25% of pediatric cancers.
- **Format: .bmp**



Class .

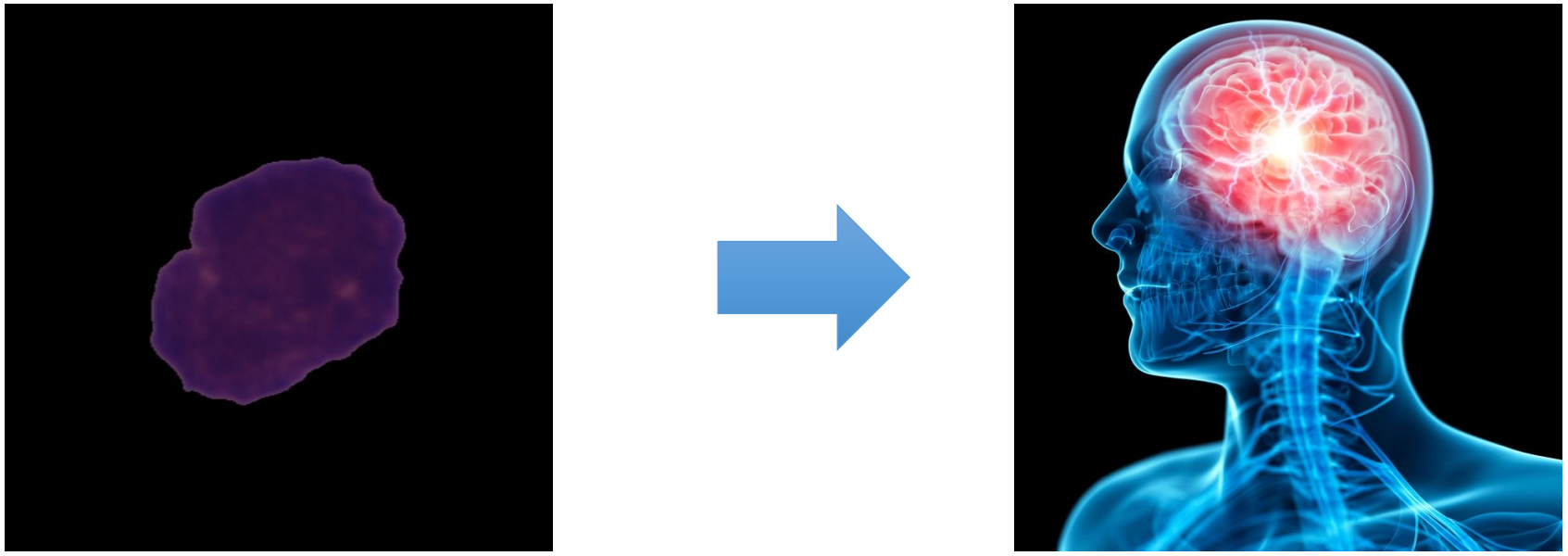
0 - Normal cell

1 - Leukemia blast

- *Reference :* Leukemia Classification | Kaggle

Prepare Data

- **7,995** images for training
- **1,599** for validating
- **1,067** for testing
- The images' resolutions are the same, which is 450 * 450 pixels.



DataLoader

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

```
class RetinopathyLoader(data.Dataset):
    def __init__(self, root, mode):

    def __len__(self):
        """'return the size of dataset"""
    def __getitem__(self, index):
        """something you should implement here"""
```

Prepare Data

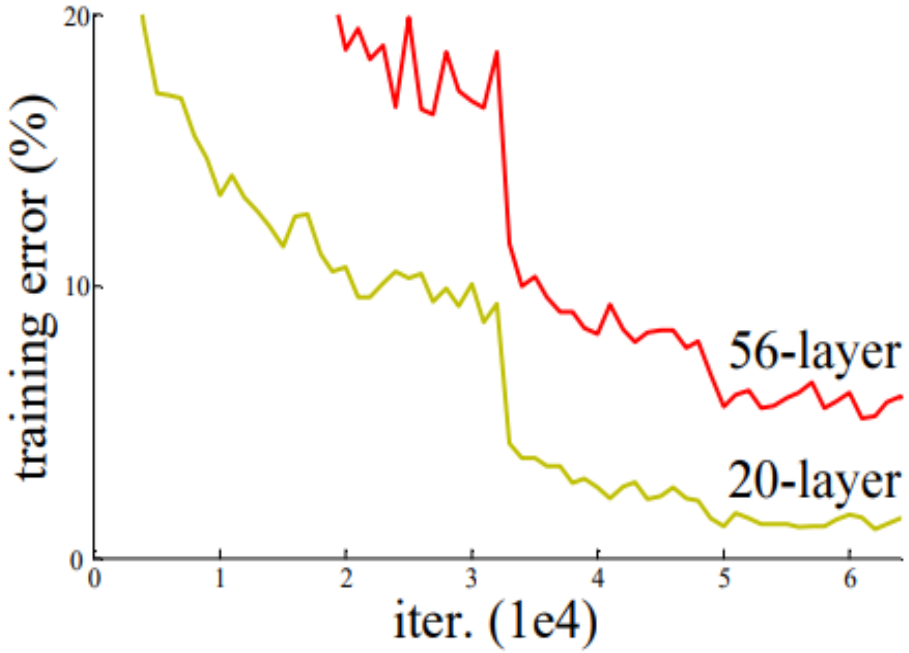
```
def __init__(self, root, mode):
    """
    Args:
        root (string): Root path of the dataset.
        mode : Indicate procedure status(training or testing)

        self.img_name (string list): String list that store all image names.
        self.label (int or float list): Numerical list that store all ground truth label values.
    """
    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)
```

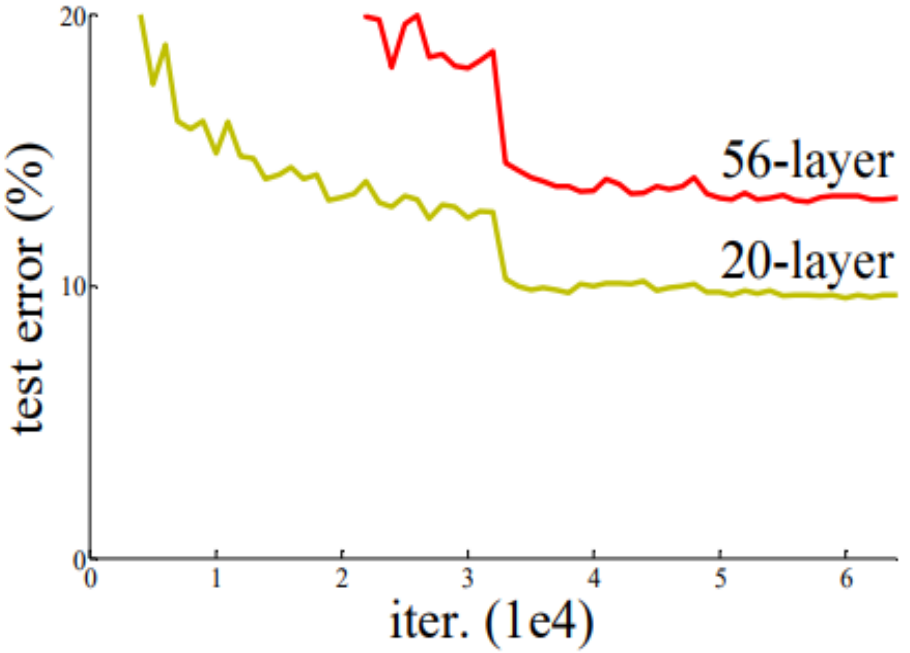
Prepare Data

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

    """
       step1. Get the image path from 'self.img_name' and load it.
              hint : path = root + self.img_name[index] + '.jpeg'

       step2. Get the ground truth label from self.label

       step3. Transform the .jpeg rgb images during the training phase, such as resizing, random flipping,
              rotation, cropping, normalization etc. But at the beginning, I suggest you follow the hints.

              In the testing phase, if you have a normalization process during the training phase, you only need
              to normalize the data.

              hints : Convert the pixel value to [0, 1]
                      Transpose the image shape from [H, W, C] to [C, H, W]

        step4. Return processed image and label
    """

    return img, label
```

ResNet

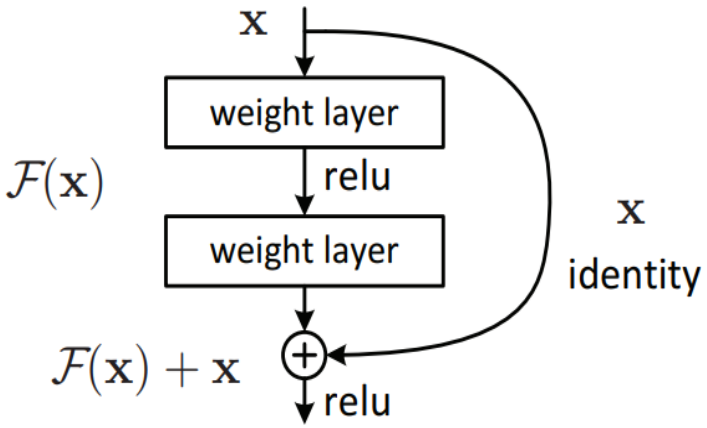
- ResNet (Residual Network) is the Winner of ILSVRC 2015 in image classification, detection, and localization, as well as Winner of MS COCO 2015 detection, and segmentation

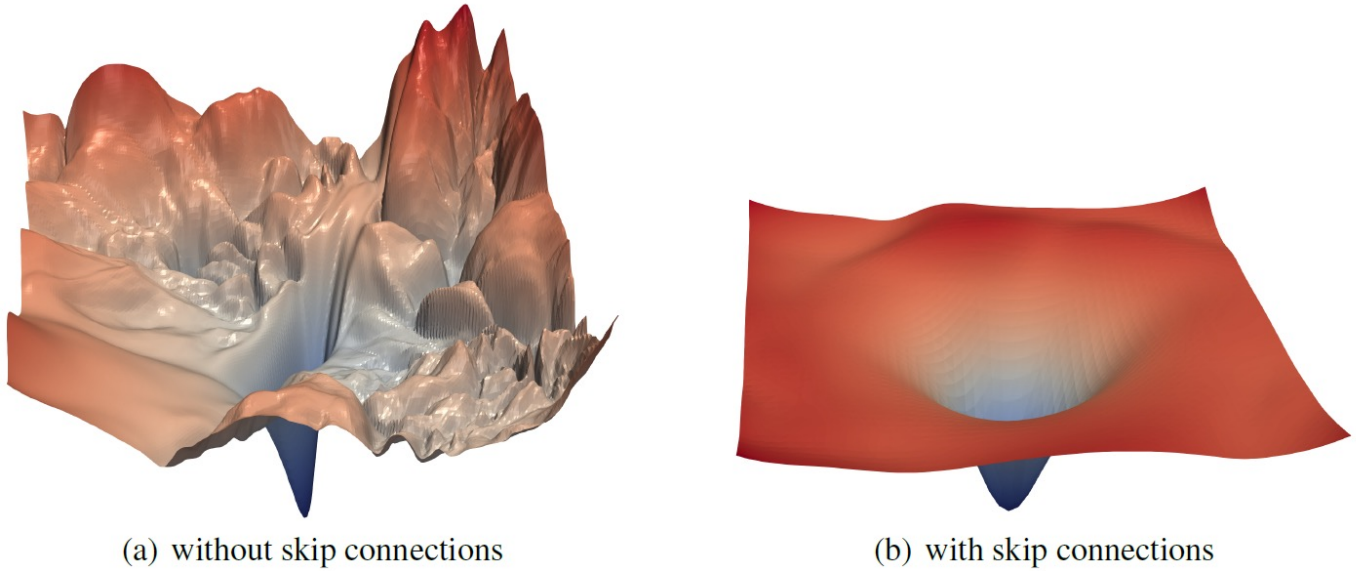




ResNet

- 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

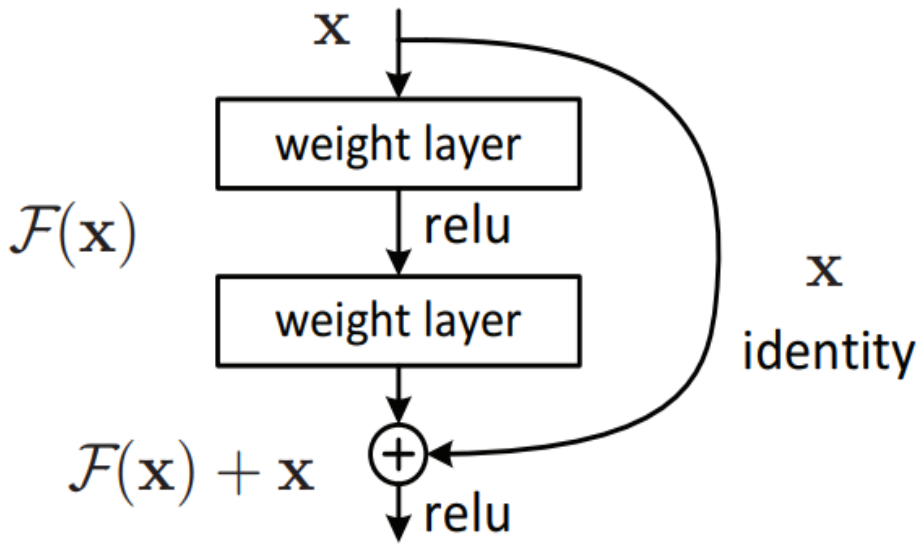
$$x \rightarrow w_1 \rightarrow y_1 \quad w_2 \rightarrow y_2 \quad w_3 \rightarrow y_3 \quad w_4 \rightarrow y_4 \rightarrow Loss$$

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

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

ResNet

- ResNet can avoid vanishing gradient problem



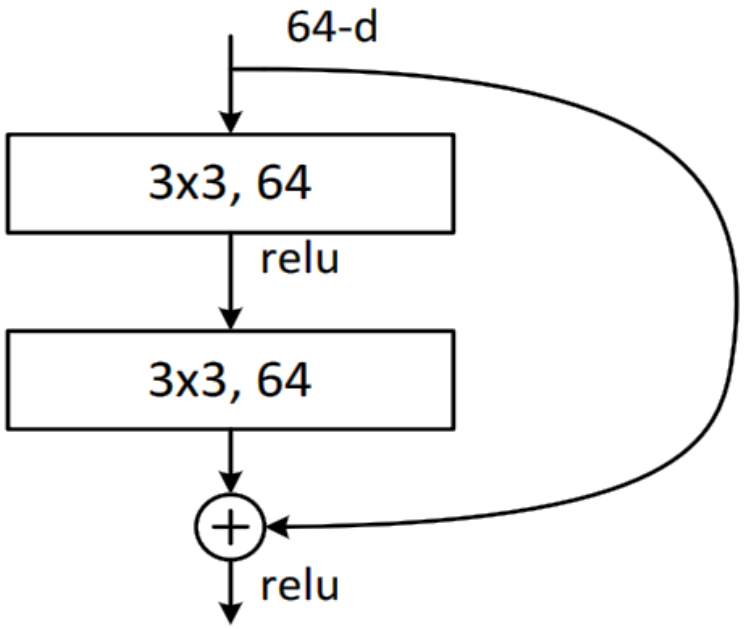
$$\mathbf{x}_L = \mathbf{x}_l + \sum_{i=l}^{L-1} \mathcal{F}(\mathbf{x}_i, \mathcal{W}_i),$$

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

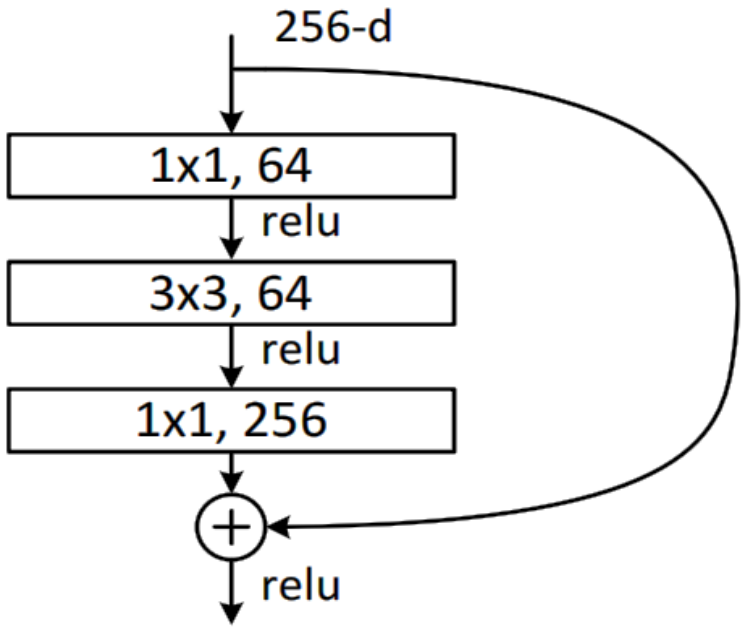
ResNet

- ResNe18 (Basic block), ResNet50 & ResNet152 (Bottleneck block)

Basic block

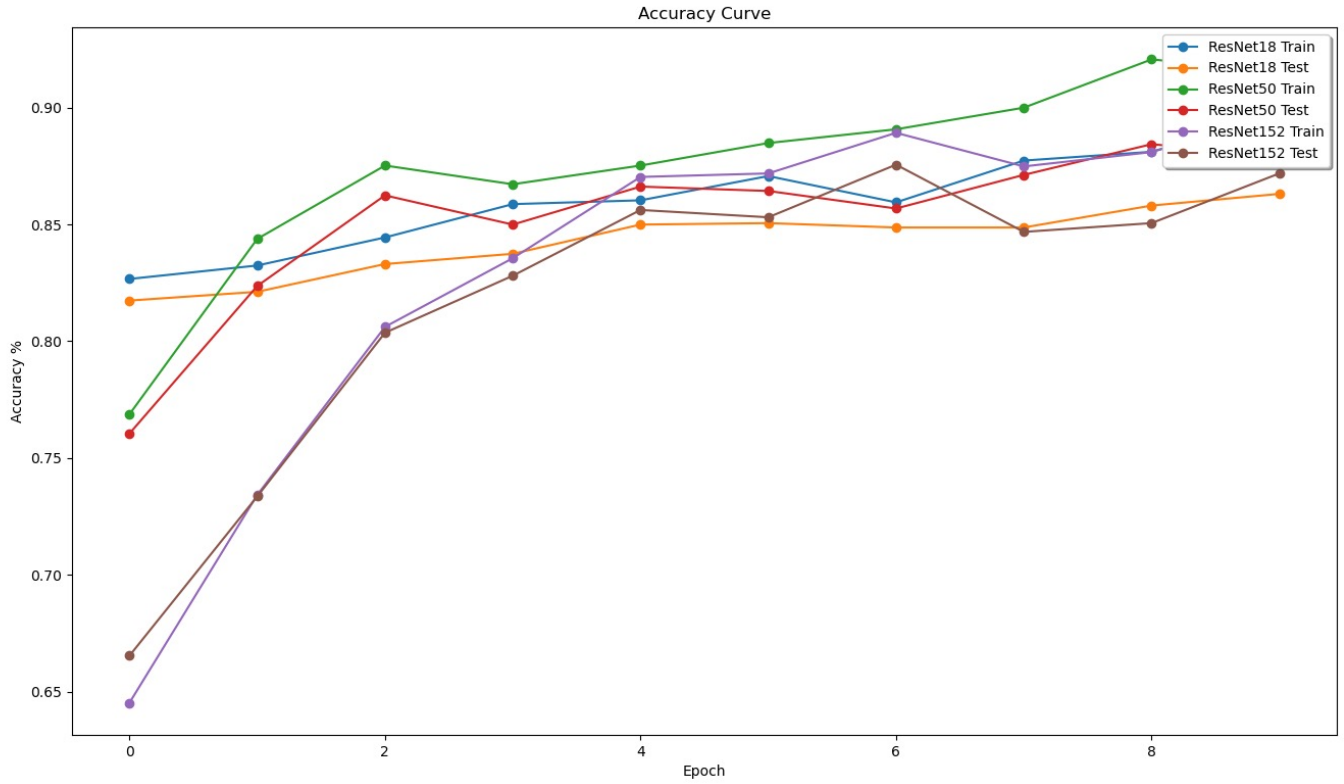


Bottleneck block



Result Comparison

- Compare and visualize the accuracy trend **between the 3 model architectures**, you need to plot each epoch accuracy (not loss) during training phase and testing phase.



Confusion Matrix

- A confusion matrix is a table that is often used to describe the performance of a classification model

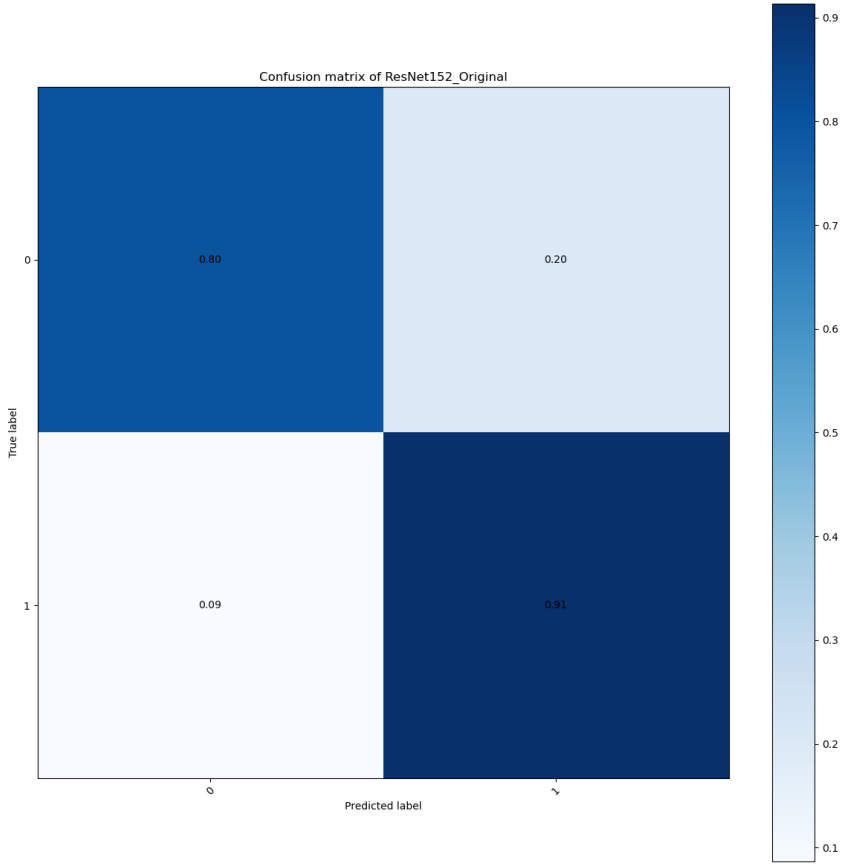
- y_true : ground truth label array
- y_pred: prediction array
- Classes: label name [‘0’, ‘1’]

```
def plot_confusion_matrix(y_true, y_pred, classes,
                          normalize=False,
                          title=None,
                          cmap=plt.cm.Blues):
```

Reference: https://scikit-learn.org/stable/auto_examples/model_selection/plot_confusion_matrix.html#sphx-glr-auto-examples-model-selection-plot-confusion-matrix-py

Confusion Matrix

- Calculate the confusion matrix and plotting



Report Spec

1. Introduction (10%)
2. Implementation Details(30%)
   A. The details of your model (ResNet)
   B. The details of your Dataloader
   C. Describing your evaluation through the confusion matrix
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
      - **(ResNet18, ResNet50, ResNet152)**
5. Discussion (30%)
   A. Anything you want to share

Kaggle Competition

Kaggle Result Format

- The result should be a .csv file
- The file should contain the image id and the predicted label, which should look like the following figure:

ID	label
./new_dataset/test/0.bmp	1
./new_dataset/test/1.bmp	1
./new_dataset/test/2.bmp	1
./new_dataset/test/3.bmp	1
./new_dataset/test/4.bmp	1
./new_dataset/test/5.bmp	0

Kaggle Evaluation

- We use Accuracy Score metric to evaluate your prediction result on the Kaggle website

- Since there are 3 different model architectures, there are 3 Kaggle competitions:
  - Leukemia Classification Competition - ResNet18
  - Leukemia Classification Competition - ResNet50
  - Leukemia Classification Competition - ResNet152

- Please set your team's name as your student ID
  - Otherwise, you'll get no points in the competition
  - Also, you'll be kicked out of the competition, and get no points

---- Criterion of result (30%) ----

For each ResNet competition (10%)

- 1st : 10 pts
- 2nd : 9 pts
- 3rd : 8 pts
- Top 25%: 7 pts
- Top 50%: 6 pts
- Others above baseline: 5 pts (Accuracy: 75%)
- Below baseline: 0 pts

- The total score is the sum of 3 competition ranking result.
- The baseline is on the leaderboard on the kaggle website

---- Notices ----

Score: 30% experimental results + 30% report + 40% demo score
P.S If the zip file name or the report spec have format error, it will be penalty (-5).

In demo phase, you need to load trained model and show your results of the 3 different models. If you failed to do so, you will get 0 pts on the model that you couldn’t successfully demonstrate.

If you’re found plagiarizing or faking your model predicting result on Kaggle, you’ll get 0 pts in this lab. Also, we will take disciplinary action in accordance with school regulations.

Please make sure the code and the model weight you uploaded can be successfully executed by the teaching assistant.