Classify Leaves - Individual Final Project Report

Hui Yang

April 27, 2022

1 Introduction

In this project, we are interested in the image classification problem of leaves. We want to use different models including a convolutional neural network (CNN) baseline that we built ourselves, more complicated CNN models like ResNet as well as the more advanced models like Vision Transformer (Vit). In this project Kurtis was focusing on the Exploratory Data Analysis, Data Augmentation, building the code structures and the report writing. He also worked on ResNet modelling. Changhong was focusing on the Vit modelling, training all models that are used in the projects and plotting the modeling results with Tensorboard. I was focusing on the CNN baseline model and deeper models like VGG-16 and visualizing the CNN baseline model.

2 Exploratory Data Analysis

The dataset is from Kaggle competition Classify Leaves. (Classify Leaves, 2021). It is a collection of leaf images, expanding 176 categories where the breakdown of classifications is shown in . There are 18353 training images, 8800 test images with each category having at least 50 images. (Classify Leaves, 2021)

3 Models

3.1 CNN baseline model

We built a Convolutional Neural Network as a baseline for our modeling whose architecture is described in Table 1. We also visualized the filters and feature maps to better understand how the CNN model works in Sec. 3.1.1. Our CNN baseline model has only 4 Conv2d layers and 2 fully connected)FC) layers and to test the effects of deeper networks, we also run the VGG-16 pretrained model for comparison which will be described in Sec. 3.1.2.

3.1.1 Visualization of CNN baseline model with filters and feature maps

Filters are the learned weights of the kernels that are used to convolve with the feature maps whereas the feature map, also called Activation Map, is obtained with the convolution operation, applied to the input data using the filter/kernel.

We plot the first layer filter for the trained CNN baseline model in Fig. 3 which has the size of [3, 16, 7, 7]. The first dimension of size 3 is represented by RGB colors shown in the figure. The second to fourth layers of filters are shown from Fig. 4 to Fig. 6. For those layers we only plot part of the whole filters since the first dimensions of 2nd/3rd/4th layers are 16, 64, 128 which are impossible to present in any color map. For the second dimension we only plot first 64 filters out of the whole 64/128/256 filters.

We also plot some feature maps of the raw image of a maclura pomifera type leaf shown in Fig. 7. Fig. 8 and Fig. 9 show the feature maps after the Conv2d and ReLU/BatchNorm/MaxPool in the first convolutional layer respectively whereas Fig 10 shows the feature maps after the second Conv2d.

The flowchart of how the first two convolutional layers work with the filters and feature maps is shown in Fig. 11.

We don't plot all other filters and feature maps in the last two layers as the channel numbers grow up in deeper neural layers and it becomes more difficult to make the plots in a meaningful way and to comprehend. This is always a dilemma in deep learning that as the model becomes more complex, the performance may get better whereas the interpretability becomes less comprehensive.

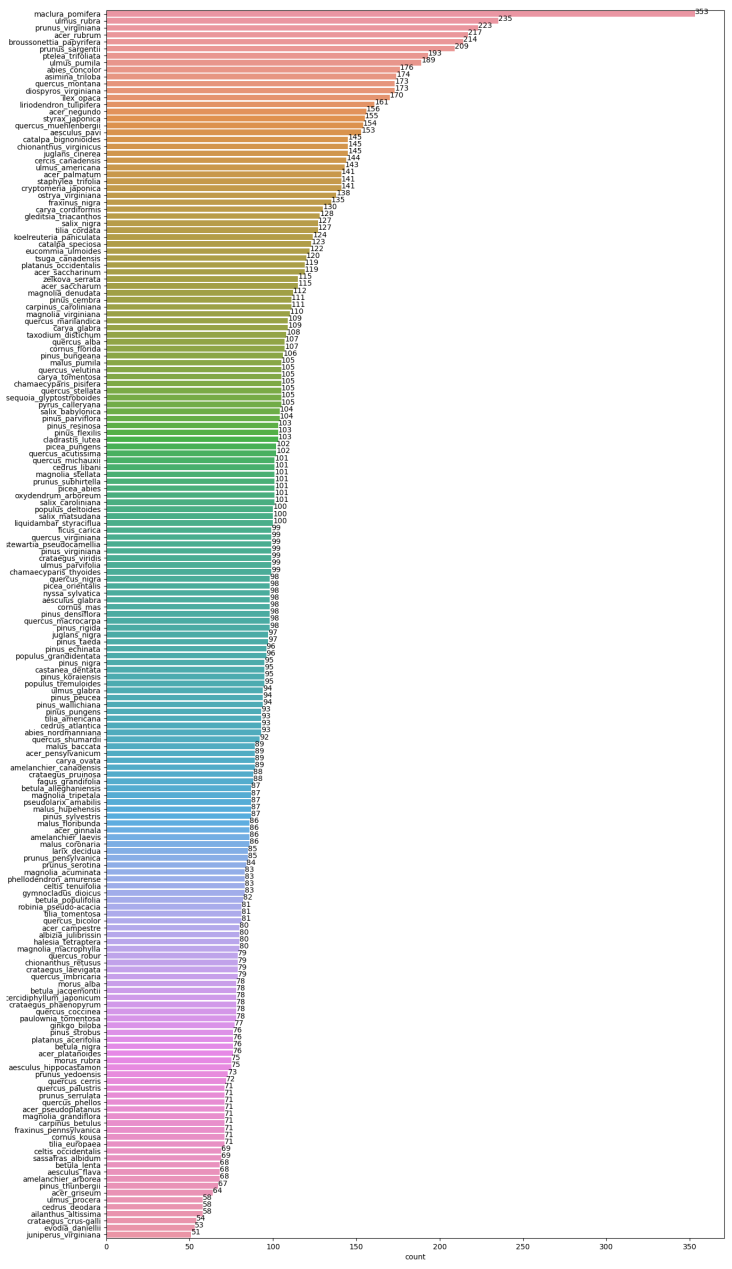

Figure 1: The class breakdown of 176 classes of leaves.

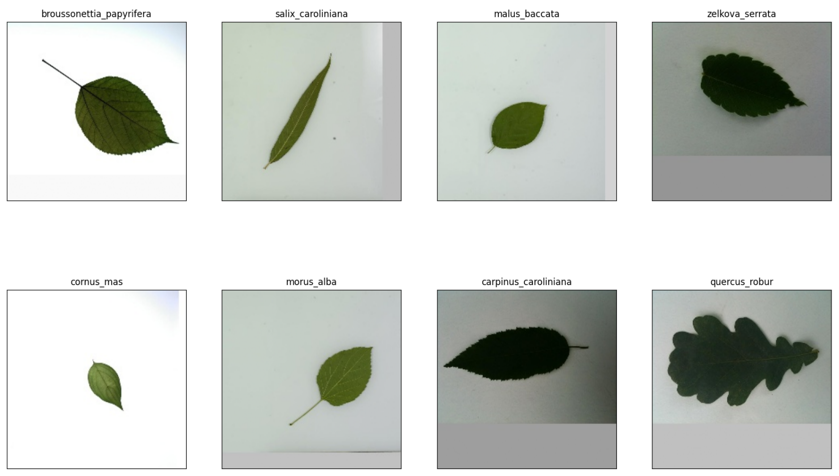

Figure 2: Some examples of leaves in our dataset.

Table 1: The CNN baseline model architecture and the Output size of each layer

Layer	Output Shape
	[3, 224, 224]
Conv2d(3, 16, kernel_size=(7, 7), stride=(2, 2))	[16, 109, 109]
ReLU	[16, 109, 109]
BatchNorm2d	[16, 109, 109]
MaxPool2d(kernel_size=(3, 3), stride=2)	[16, 54, 54]
Conv2d(16, 64, kernel_size=(5, 5), stride=(2, 2))	[64, 25, 25]
ReLU	[64, 25, 25]
BatchNorm2d	[64, 25, 25]
MaxPool2d(kernel_size=(3, 3), stride=2)	[64, 12, 12]
Conv2d(64, 128, kernel_size=(3, 3), stride=(1, 1))	[128, 10, 10]
ReLU	[128, 10, 10]
BatchNorm2d	[128, 10, 10]
Conv2d(128, 256, kernel_size=(3, 3), stride=(1, 1))	[256, 8, 8]
ReLU	[256, 8, 8]
AdaptiveAvgPool2d	[256, 1, 1]
Linear	[4096]
ReLU	[4096]
Linear	[176]
BatchNorm1d	[176]

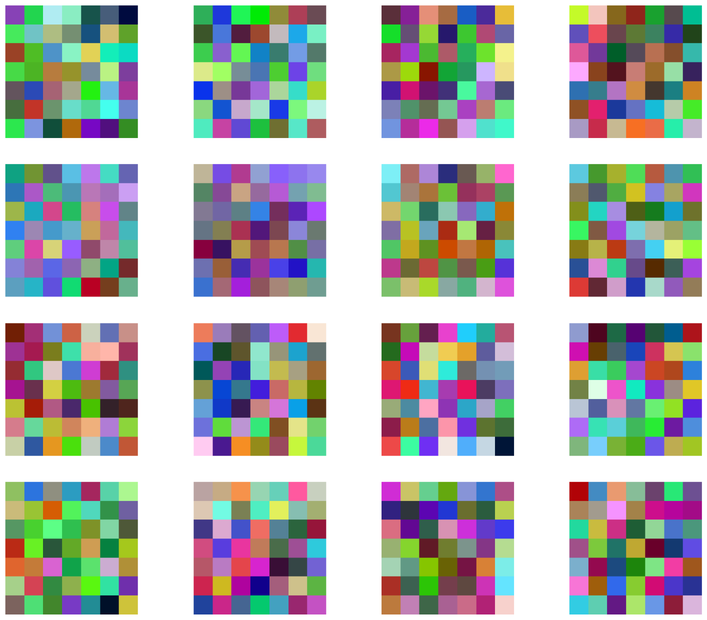
Figure 3: The filters of the first layer kernels with size of [3, 16, 7, 7]

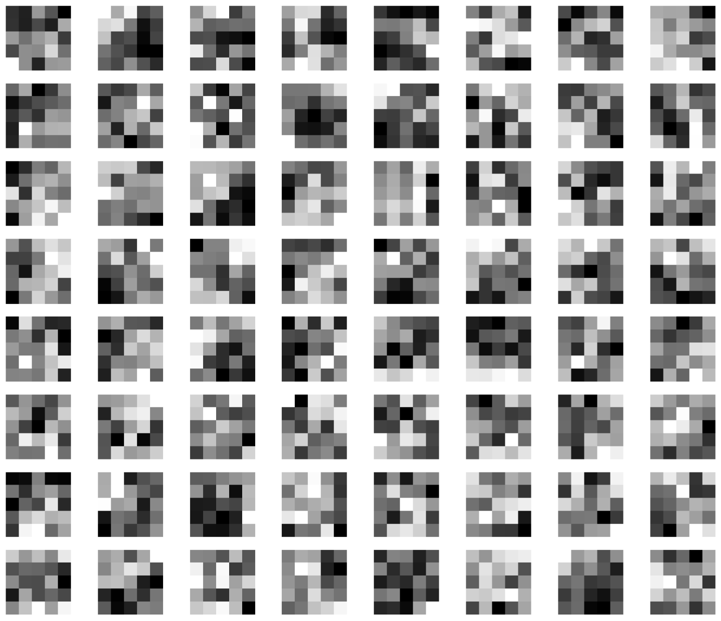
Figure 4: The first of the 16 filters of the second layer kernels with size of [16, 64, 5, 5]

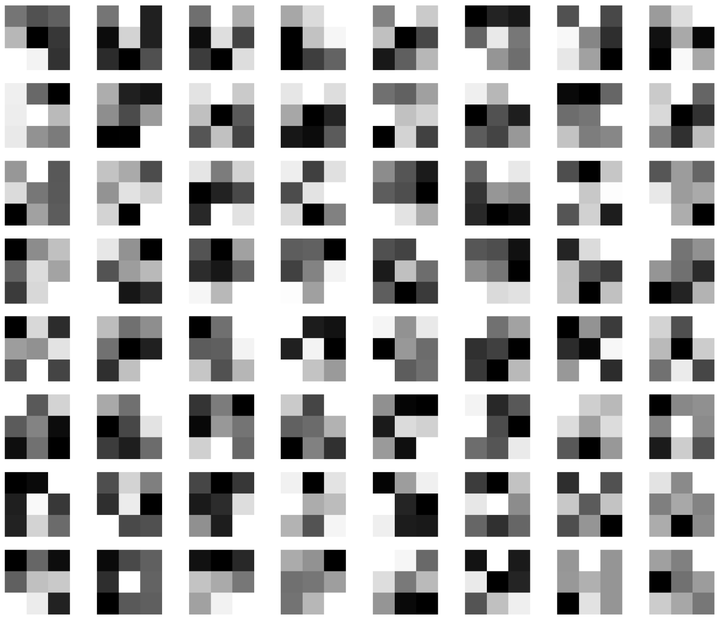

Figure 5: The first 64 filters of the 64x128 filters of the third layer kernels with size of [64, 128, 3, 3]

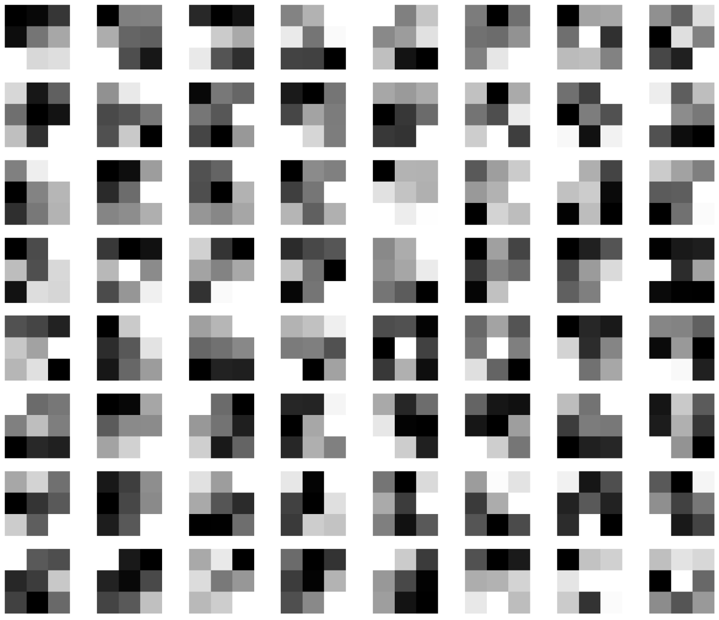

Figure 6: The first 64 filters of the 128x256 filters of the third layer kernels with size of [128, 256 3, 3]

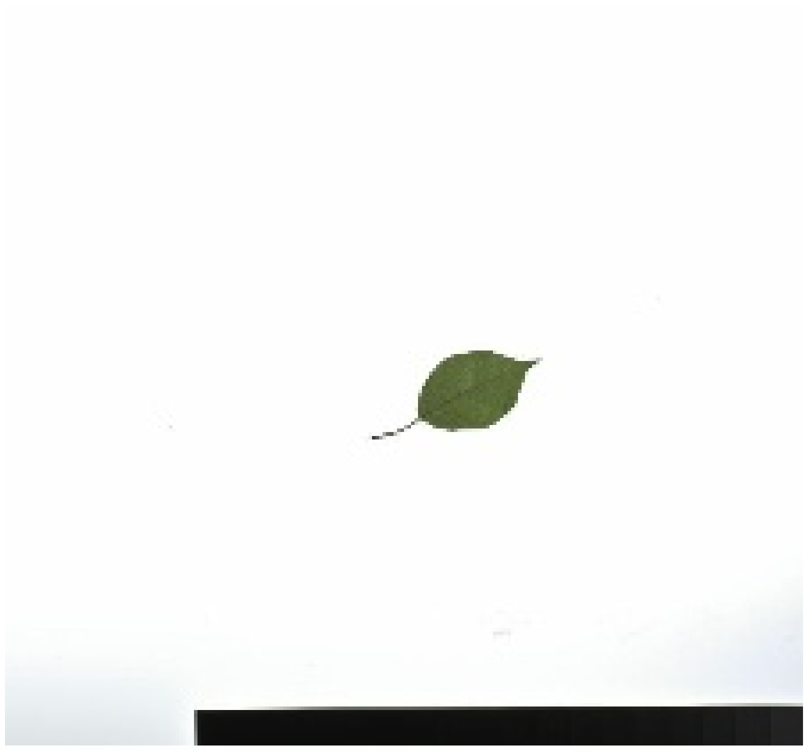

Figure 7: The raw image of a maclura pomifera type leaf used for feature maps

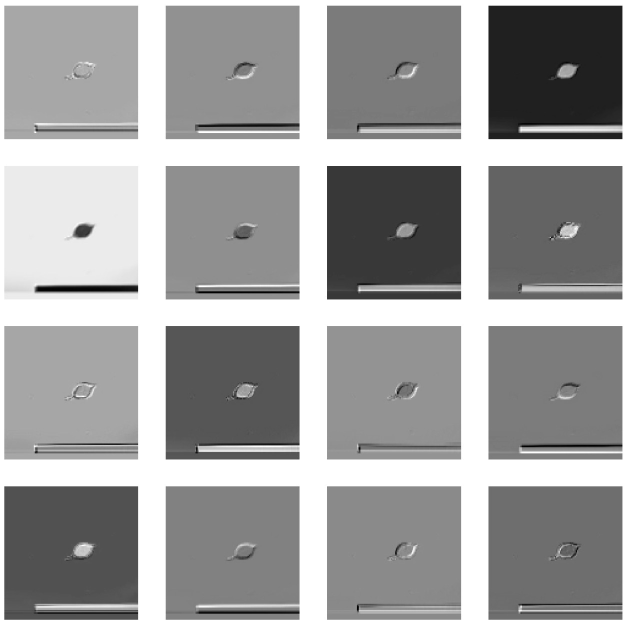

Figure 8: The feature maps after Conv2d in the first convolutional layer

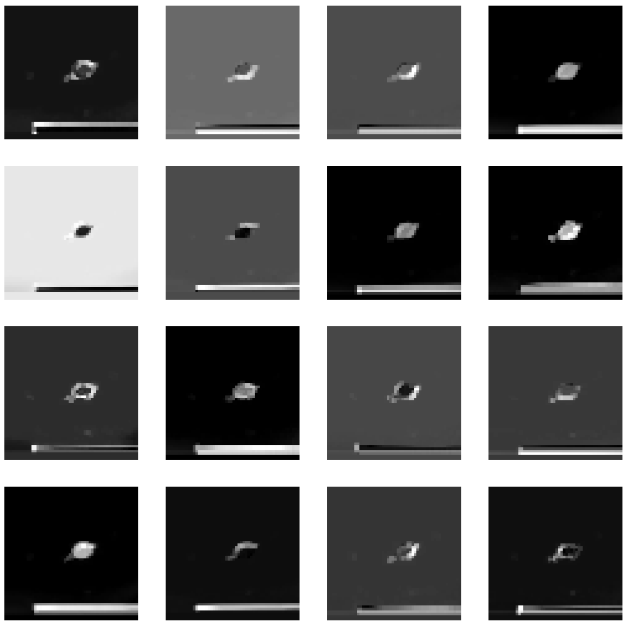

Figure 9: The feature maps after ReLU, BatchNorm and MaxPool in the first convolutional layer

3.1.2 A very deep CNN model: VGG-16

VGG-16 (Very Deep Convolutional Networks For Large-Scale Image Recognition) Simonyan and Zisserman, 2014

To take another step further to test a deeper CNN model, we used VGG16 (Very Deep Convolutional Networks For Large-Scale Image Recognition) Simonyan and Zisserman, 2014 on our dataset and compared its performance with our CNN baseline model. The VGG-16 model has a total number of 13 convolutional layers and 3 fully connected layers where the architecture is shown in Table 2.

The comparison of performance of the baseline model and the VGG-16 model is shown in Table 3. We see that VGG-16 doesn't perform better than the CNN baseline model while the depth and the complexity of the VGG-16 model is much higher. This may relate to the degradation problem that even for a deeper network starts to converge, the accuracy gets saturated and degrades as the network depth keeps increasing. This leads to our consideration of the ResNet models which will be introduced in the next section.

3.2 ResNet

Recent works have shown that image classification tasks can benefit from very deep models. However, the vanishing/exploding gradients problem may come with a deeper network. This problem has been largely mitigated by the normalization initialization and intermediate normalization layers, which enable networks with tens of layers to start converging with stochastic gradient descent (SGD). Another problem, called the degradation problem, has been exposed, as deeper networks start to converge, where the accuracy gets saturated and then degrades rapidly as the network depth keeps increasing. The ResNet model was proposed to address the degradation problem by introducing a deep residual learning framework He et al., 2015. The building block of the residual learning can be realized by feedforward neural networks with "shortcut connections" (see Fig. 13). In ResNet models, the shortcut connections simply perform identity mapping, and their outputs are added to the outputs of the stacked layers. Identity shortcut connections add neither extra parameter nor computational complexity. There are a number of Resnet variants with different numbers of layers. And here we plot the model architecture of one of them – the ResNet18 model in Fig. 12 which performs the best among all the models that we have tried in this project.

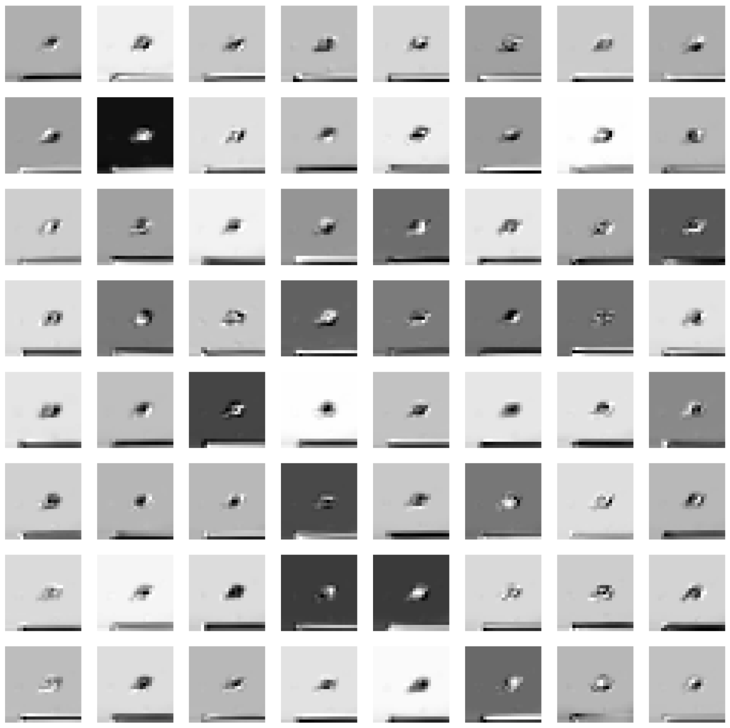

Figure 10: The feature maps after Conv2d in the second convolutional layer

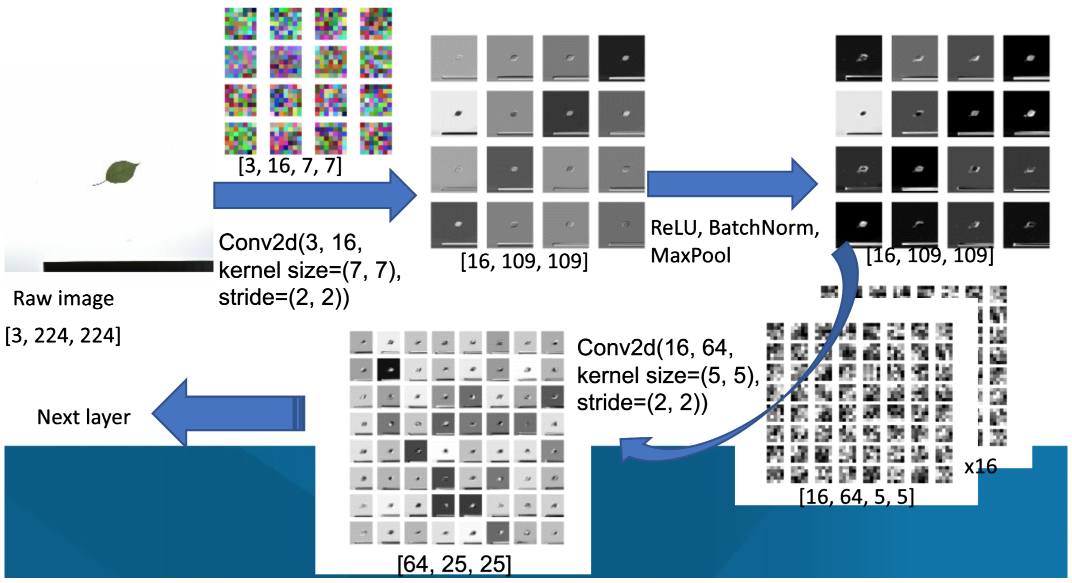

Figure 11: the flowchart of how the first two convolutional layers operate with the filters and feature maps

Table 2: VGG-16 baseline Architecture

Layer	Output Shape
	[3, 224, 224]
Conv2d(3, 64)	[64, 224, 224]
ReLU	[64, 224, 224]
Conv2d(64, 64)	[64, 224, 224]
ReLU	[64, 224, 224]
MaxPool2d	[64, 112, 112]
Conv2d(64, 128)	[128, 112, 112]
ReLU	[128, 112, 112]
Conv2d(128, 128)	[128, 112, 112]
ReLU	[128, 112, 112]
MaxPool2d	[128, 56, 56]
Conv2d(128, 256)	[256, 56, 56]
ReLU	[256, 56, 56]
Conv2d(256, 256)	[256, 56, 56]
ReLU	[256, 56, 56]
Conv2d(256, 256)	[256, 56, 56]
ReLU	[256, 56, 56]
MaxPool2d	[256, 28, 28]
Conv2d(256, 512)	[512, 28, 28]
ReLU	[512, 28, 28]
Conv2d(512, 512)	[512, 28, 28]
ReLU	[512, 28, 28]
Conv2d(512, 512)	[512, 28, 28]
ReLU	[512, 28, 28]
MaxPool2d	[512, 14, 14]
Conv2d(512, 512)	[512, 14, 14]
ReLU	[512, 14, 14]
Conv2d(512, 512)	[512, 14, 14]
ReLU	[512, 14, 14]
Conv2d(512, 512)	[512, 14, 14]
ReLU	[512, 14, 14]
MaxPool2d	[512, 7, 7]
AdaptiveAvgPool2d	[512, 7, 7]
Linear	[4096]
ReLU	[4096]
Dropout	[4096]
Linear	[176]

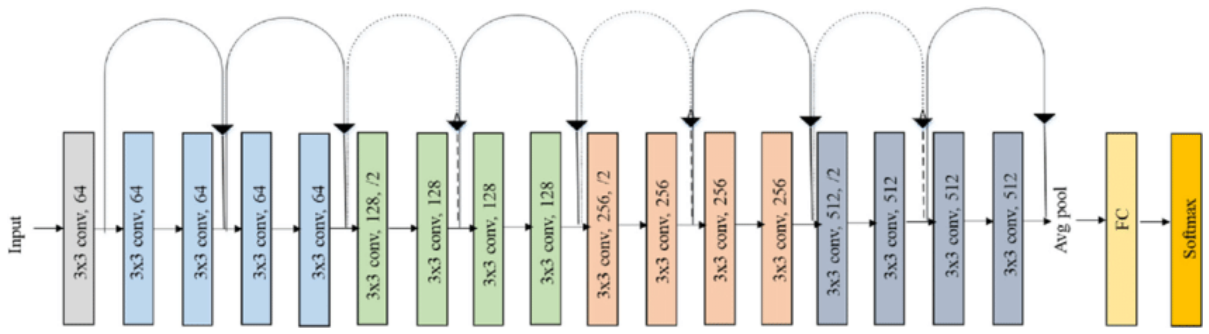

Figure 12: The architecture of Resnet18 model. from He et al., 2015

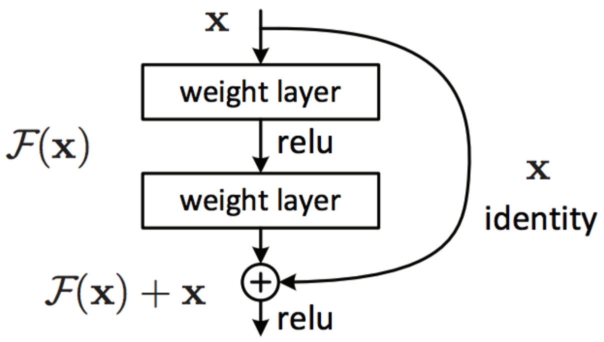
x
weight layer
relu
weight layer
$\mathcal{F}(\mathbf{x})$
x
identity
$\mathcal{F}(\mathbf{x}) + \mathbf{x}$
relu

Figure 13: The building block of Residual learning from Ramzan et al., 2019

Table 3: Model Performance

Models	Epochs	Batch size	Learning rate	accuracy	f1_macro	coh	sum
CNN-baseline	20	50	3e-04(mode='min', factor=0.5, patience=5)	0.846	0.840	0.845	2.530
VGG-16	20	50	3e-04(mode='min', factor=0.5, patience=5)	0.826	0.822	0.825	2.472

4 Results

- The CNN baseline model with 4 convolutional layers and 2 FC layers achieves an accuracy of 84.6% while the VGG-16 model only achieving an accuracy of 82.6% which is much deeper than the CNN baseline model with 13 convolutional layers and 3 FC layers. This may relate to the degradation problem as we increase the depth of the CNN models.
- I made the filters and feature maps of the CNN baseline model to help understand how CNN model works and seek for the interpretability of the modeling. However, as the model gets more complicated, the filters and feature maps grow their sizes and become more difficult to comprehend.
- To mitigate the degradation model, the ResNet model was proposed by other works, introducing a deep residual learning framework. We tested ResNet-18 in our project which achieves the best performance among all models in our project and the details of the modeling performance will be covered in Changhong's report as well as the comparison to other models.

5 Percentage of code found or copied from the internet

The modelling part of the code is mostly from the template code from midterm II. The visualization part of the code has about 50% copied from the internet.

References

Simonyan, K., & Zisserman, A. (2014). Very Deep Convolutional Networks for Large-Scale Image Recognition. arXiv e-prints, Article arXiv:1409.1556, arXiv:1409.1556.

He, K., Zhang, X., Ren, S., & Sun, J. (2015). Deep Residual Learning for Image Recognition. arXiv e-prints, Article arXiv:1512.03385, arXiv:1512.03385.

Ramzan, F., Khan, M. U., Rehmat, A., Iqbal, S., Saba, T., Rehman, A., & Mehmood, Z. (2019). A deep learning approach for automated diagnosis and multi-class classification of alzheimer's disease stages using resting-state fmri and residual neural networks. Journal of Medical Systems, 44. https://doi.org/10.1007/s10916-019-1475-2