CPSC 340 Machine Learning Take-Home Final Exam Q3

Minsi Sung	Nourhan Bayasi	Tam Nguyen	Son Thai (Tyson) Tong
ID: 44103232	ID: 23878614	ID: 61409165	ID: 20320644

1 Abstract

The objective of this research paper is to implement **transfer learning** that use pre-trained models for making a multi-class classifier to classify chest X-ray images into three classes; COVID-19, Pneumonia, and Normal. The data was collected from two main sources, and we ended up having 20,092 images for training, 5,970 for validation, and 280 for testing. The pre-trained models for transfer learning in this research paper are ResNet18 and VGG16 which were trained by ImageNet. Four different methods to adjust the part of pre-trained model architectures with tuned hyperparameters are provided for checking performance and comparison. VGG16 pre-trained network with BCE and Adam optimizer performed the best, with an average accuracy of 90%.

2 Introduction

On March 11, the World Health Organization declared the rapid spreading Coronavirus disease 2019 (COVID-19) is considered a pandemic that was characterized by the rapid and global spread around the world. As new confirmed and death cases rapidly increase around the world, quick and accurate COVID-19 detection systems are needed to lower the chance of COVID-19 virus spreading among people. In this research paper, four different transfer learning models based on ResNet18 and VGG16 are proposed. Chest X-ray images are used for training the models to classify images into three different classes which are COVID-19, Pneumonia including "SARS", "MERS", "ARDS", "Streptococcus", "Pneumonia", "Viral Pneumonia", and "Bacterial Pneumonia", and finally Normal.

Transfer learning is a technique in which a neural network model is first trained on a problem similar to the new target problem. One or more layers from the trained model are then used in a new model trained on the problem of interest. It has the benefit of decreasing the training time for a neural network model and able to achieve lower test error compared to the task model that is trained on relative small amount of dataset. In this research paper, ResNet18 and VGG16 are chosen for transfer learning to construct the desired multi-class classifier. These two models are trained on ImageNet which has more than 1,000,000 images to classify 1,000 categories. They have learned how to detect generic features from photographs. Note that convolution layers that are closer to the input layer of the model learn low-level features, layers in the middle of the layer learn more complex abstract features that combine the lower level features extracted from the input, and layers closer to the output interpret the extracted features in the context of a classification task. Since our goal is similar to pre-trained models to classify images, the output from much deeper layers in the model can be used. Following the reason mentioned above, two of well-known transfer learning models are used and adjusted to classify our data, and more details are given next.

3 Methods

3.1 Architectures

The architecture of VGG16 consists of 16 weight layers (3×3 convolution filters) plus 3 full-connected layers before soft-max layer to classify 224×224 RGB input images into 1000 categories (More in detail can be found at: https://arxiv.org/abs/1409.1556). For ResNet18, the architecture inserts the shortcut connection turning the network into residual version to ease the training of networks that are substantially deeper. The last two layers of architecture are: an average pool layer and a fully-connected layer. The authors provide comprehensive empirical evidence showing that these residual networks are easier to optimize, and can gain accuracy from considerably increased depth (More in detail can be found at: https://arxiv.org/abs/1512.03385).

ResNet18 and VGG16 took days for training and have learned how to detect generic features from images. Note that convolutional layers that are closer to the input layer of the model learn low-level features, layers in the middle of the layer learn more complex abstract features that combine the lower level features extracted from the input, and layers closer to the output interpret the extracted features in the context of a classification task. Since our goal is similar to the goal of ResNet18 and VGG16 to classify images, output from deeper layers of these two pre-trained models can be used through transfer learning to avoid the long training time for a neural network model.

3.2 Data Collection and Preparation

The dataset used to train, evaluate and test the networks is comprised of a total of 26,062 X-ray images collected and combined from two publicly available datasets. COVID-19 + cases are collected from: https://github.com/ieee8023/covid-chestxray-dataset.git, which has a public open dataset of chest X-ray and CT images of patients which are positive or suspected of COVID-19. For COVID-19 − cases, the data is collected from the Kaggle RSNA Pneumonia Detection Challenge dataset from: https://www.kaggle.com/c/rsna-pneumonia-detection-challenge/data. In addition, the training and the testing data lists given in https://github.com/lindawangg/COVID-Net/blob/master are used. The X-ray image distribution for each class is listed in DataPreparation.ipynb file. Furthermore, it has been noticed that there was a limited amount of COVID-19 infected cases and the associated X-ray images. Thus, we augmented the COVID-19 images. We used different cropping and flipping of the COVID cases (Dr. Cohen's GitHub data) to make the number larger. Also, as part of the data preparation, we resized all images to 224×224×3 for use in ResNet18 and VGG16, and subtracted the mean values which were (RGB): 131.381, 131.03, 130.698.

4 Experiments and Results

As mentioned earlier, we have taken the advantage of ImageNet; and the state-of-the-art architectures pre-trained on ImageNet dataset. That is, instead of random initialization, we initialize the networks with a pretrained one and fine-tuned it with the training set. To implement the transfer learning on our data, we started by using Tensorflow. However, when we tested the code, there was an error due to memory allocation. The error was coming from the data shuffling command; and it was "*Unable to allocate array with shape and data type*". We tried to debug and fix the error but because it was something related to the system's overcommit handling mode, we couldn't do it. Then, we re-implemented the code in Pytorch, and we used basically DataLoader, allowing by so sending small units in batch size to GPU for gradual computations and hence allowing to bypass memory error as it doesn't attempt to load all data at once.

We have done four experiments in total; two for each network. For the first two, we freezed (kept weights) of the pre-trained model layers (ResNet18 and VGG16 respectively) except the last layer which is softmax layer, because in our implementation we have 3 labels only and not 1000 as in ImageNet. For the other two experiments, we fine-tuned more layers. Specifically, for ResNet18, the last two layers, which are the average pooling layer and softmax layer, were tuned. In case of VGG16, we trained the last three fully-connected layers and softmax layer. In each experiment, we tried two optimizers with BCE loss; SGD and

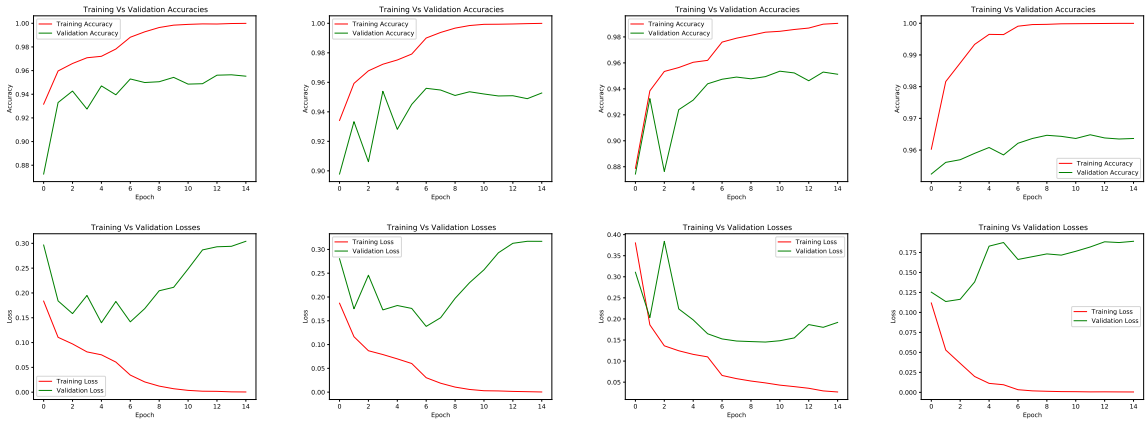

Figure 1: Top: Accuracy of Resnet18 with Adam, Accuracy of Resnet18 with SGD, Accuracy of VGG16 with Adam, Accuracy of VGG16 with SGD. Bottom: Loss of Resnet18 with Adam, Loss of Resnet18 with SGD, Loss of VGG16 with Adam, Loss of VGG16 with SGD

Accuracy per Class					
		Normal	Pneumonia	COVID-19	Average
ResNet18	Adam Optimizer	94%	92%	66%	85%
	SGD Optimizer	90%	94%	80%	88%
VGG16	Adam Optimizer	96%	93%	80%	90%
	SGD Optimizer	96%	90%	75%	87%

Figure 2: Test accuracy of pre-trained ResNet18 and VGG16 models with different Optimizers

Adam. We used a decay learning rate by a factor of 0.1 every 3 epochs (initial value was set to be 0.01 after trying 0.1, 0.01, and 0.001), and we run the code over a total of 15 epochs and minibatch size of 15 (also a hyperparameter). We found out that the results from the third and fourth experiments were much worse than the results from the first two experiments, as we got an accuracy between 60% and 75% compared to an accuracy between 85% and 90% respectively. Thus, only results from experiments 1 and 2 (*Fine-tuning the last layer only of ResNet18 and VGG16*) are highlighted.

The training and validation accuracy and loss of each pre-trained network on our data are shown in Figure 1. As shown, we noticed that we were overfitting the network after epoch = 6, since the validation loss has started to go up, so we repeated the experiments with a new epoch value (epoch = 6). After that, we computed the accuracy of the network on the test images and the results are shown in Figure 2. VGG16 pre-trained network with BCE and Adam optimizer performed the best, with an average accuracy of 90%. Also, we can tell from the results that the accuracy of detecting COVID-19 is not promising as the other two classes, and that's because our COVID-19 data is not as generalized as the other two.

5 Conclusion

In this rearch paper, we have worked on a COVID-19-related machine learning classification task and attempted to solve it using pre-trained well-known networks. COVID-19 data augmentation was done to improve the balance in the dataset. After multiple experiments and hyperparameters tunning, we got a classification accuracy of 90% using VGG16 with fine-tuning the last layer.