Replies to critical reviews

Critical review from team 8:

1. The VGG19 model achieved really high accuracy even without fine tuning. Is there a possible explanation for it?
2. Would be clearer if the result is more thoroughly discussed. For example, for each model used, which category is the best classified and which category is the worst?

Our response:

1. It might be possible that the pretrained weight in VGG19 is suitable for our dataset.
2. If space allows, we will add them into the discussion section in the final report.

Critical review from team 12:

1. The project has used different models to train the dataset and it is better to make some explanation about which model perfors best and the reason for this.
2. The dataset used to train the mode is not big enough and the testing dataset is also small. Is it better to use more data to train the model?
3. It seems that you have used so many models, but it is better to concentrate on some of them and to make a more detailed discussion over these?
4. Could you have more detailed information about your fine-tuned model and explain why your fine-tuned models have a better result.
5. Could you make a comparison between the training efficiency between the different models?
6. In your presentation, it seems that you do not introduce the loss function and it is better to establish a more detailed loss function mechanism over this?
7. Why is the loss as a justification metric for the model and whether there is any other metric that can evaluate the models better?
8. In the literature review, the accuracy of different models over the dataset is pretty high, but in your project, the accuracy becomes lower. Could you please explain the reason behind this?
9. In your presentation, you have used deep learning and machine learning methods. Could you make a comparison between these two methods over their mechanism and their shortcomings and advantages.
10. In your presentation, you have used 7 models to train your dataset. Could you please make a comparison between the models over their structure and so on to make a conclusion over which model is more suitable for image classification in theory?

Our response:

1. In our ppt page 15, we show that VGG19 after fine-tuning is the best. In the presentation, we introduce the features of each model. We think the features of VGG19 might be suitable for this dataset.
2. We think it might be better if we have more images but since there are two classes in our dataset that only have approximately 1000 images and we are worried that it will make the model imbalanced, we choose to make each class have the same number of images in the end.

3. Due to limited time, we think that the first priority is to find the best model which fits well to our dataset and if time allows, we will concentrate on maybe the top two models.
4. Before we fine-tune the model, every layer before fully connected layers has fixed parameters which are pretrained weights originally for the Imagenet dataset. If we only tune the parameters of fully connected layers, it might be the bottleneck for the performance of our models. But, if we tune all parameters for our dataset, the model will have improved performance since the parameters are for our dataset.
5. The training times of different models seem similar since we do not need too many epochs to let the validation loss saturate. However, too complicated models, such as DenseNet and ResNet do not perform as well as VGG19. We think that it is because the VGG model is simple to avoid the model from overfitting.
6. We use Categorical Cross-Entropy loss as our loss function which is widely used for multi-class classification.
7. Loss function is a way to evaluate your model and see if your model is good or not in a numeric way.
8. In the presentation, we have said that their models only distinguish three categories and in order to reduce false negative rate, we want to distinguish more categories.
9. In our opinions, CNNs stands out because the models look at local and global features and it sees images in a structured way which is similar to the human visual perception of recognizing things.
10. In our opinion, all models have their own features and each feature might be good for different datasets.

Critical review from team 16:

1. For the fine tuning, what parameters are you trying to tune? You can specify them instead of only putting the comparison results.
2. Several ML/DL models have been mentioned, but can you specify the parameters you choose such as what is the “k” for k-nearest neighbors, what loss function or optimization method you choose for those DL models,etc?
3. In the literature review, the accuracy is high but it becomes low for your results. Could you explain the reason behind this?
4. The presentation already contains 7 models, and your further work is going to try more models. I think it’s better to concentrate on 2-3 three popular models on this specific task and focus on which parameters have the best performance. Trying too many models is indeed unnecessary.
5. Can you make a comparison between ML and DL models about their mechanism and performance?
6. Instead of trying too many models, you can also talk about some problems or issues you encountered (eg, overfitting) when you implemented some of the models and how you solved them.

Our response:

1. We are tuning the parameters which are pretrained weights originally for Imagenet dataset.
2. Our k is 8 and we use Categorical Cross-Entropy loss as our loss function which is widely used for multi-class classification. Also, we mainly use Adam and RMSprop as our optimizers.

3. It might be the reason that their models only distinguish three categories and ours are four. Or it might be possible that they have more time to fine-tune the parameters or build different fully-connected layers.
4. Yes, we have concentrated on the models with best performance and further improved it in the end.
5. CNNs sees images in a structured way which is similar to the human visual perception of recognizing things. ML models analyze images in one dimension so it might have worse performance.

6. At first, we tried to build a custom model, but we encountered the problem of gradient vanishing. Therefore, we use a simpler model to train the dataset. However, we only get about 80% of accuracy. Since we do not have that much data, we adopted the strategy of transfer learning which trains the weights in a large ImageNet dataset.

GROUP 19: CHEST X-RAY CLASSIFICATION FOR COVID-19 DETECTION

Chih-Chieh Chien, Cheng-Yu Chen, and Yun-Yi Lin

University of California San Diego, La Jolla, CA 92093-0238

ABSTRACT

Since December 2019, the Covid-19 pandemic has caused millions of deaths worldwide. In order to reduce the severity, we develop a way to accurately and rapidly diagnose a patient in a short time. We first collect the chest X-ray dataset and then use machine learning and deep learning, especially based on Convolutional Neural Network (CNN), to build the various models. They are K-nearest neighbor, Random forest, VGG19, ResNet, DenseNet121, InceptionV3, and Xception. Through these models, we can identify Covid-19 among the images and compare their accuracy. The results show that the best model is VGG19 with fine-tunning, in which the accuracy can achieve 92.25%.

***Index Terms*—** Machine Learning, Convolutional Neural Network, Deep Learning, Transfer Learning, Classification, Covid-19, CT diagnosis

1. INTRODUCTION

Covid-19 impacts the world a lot, including the economy, lives, and politics. It is an extremely contagious and fatal disease; therefore, we need to develop a fast way to diagnose this disease. However, nowadays, we still spend a few days or longer time to confirm whether a person has Covid-19 or not. In order to save much time and further stop the expansion of Covid-19, we want to use chest X-ray images to identify whether a patient is infected by Covid-19. Besides, some studies show that a chest x-ray cannot accurately distinguish between COVID-19 and other respiratory infections. Therefore, we develop a program by using machine learning and derived CNN models to identify not only Covid-19 but also chest-related diseases; specifically, we collect chest X-ray images and build various models to compare the diagnosable accuracy. To achieve this proposal, besides regular machine learning, we also focus on CNN models that have great performance on classifying images and takes less than one second to predict the result. In the process, we input chest X-ray images and send them to these derivative CNN models to make the prediction.

2. RELATED WORK

It has been over one year since the Covid-19 pandemic outbreak. There are many studies about using deep learning methods to detect Covid-19. After searching some papers, we find out that CNN models have the best performance on classifying the chest X-ray images and they all use transfer learning. In the work of Muhammad E. H. Chowdhury, he used data augmentation, translation, and rotation, because of a lack of data and transfer learning to improve the accuracy. He tried many CNN models and compared their performance and his result shows data augmentation improves 0.2% accuracy and DenseNet201 is the best[1]. Furthermore, Antonios Makris tried transfer learning with fine-tuning on many CNN models and his result shows that with fine-tuning, VGG16 has the best performance[2]. In the work of Hoon Ko, they also tried many CNN models and with two data augmentation including rotation and zoom and their result shows that ResNet-50 showed excellent diagnostic performance[3]. However, all their models only classify three categories and there is a report showing that a chest X-ray cannot accurately distinguish between Covid-19 and other respiratory infections. So, in our project, we want to classify more categories.

3. DATASET AND FEATURE

On Kaggle, people have collected and regularly updated the dataset which we will use in this task. There are four classes in this dataset: Normal, Covid-19, Lung Opacity, Viral Pneumonia shown in Figure 1. In order to make our model balanced, we decide to make every class have the same number of samples. So, we use 4000 images and 1000 for each class. We split the images into 80% (training), 10% (validation), and 10% (testing). The resolution of the original images is 299*299*3, which means they're color images and (length, width) = (299, 299). We resize the images to 224*224*3 to our deep learning models. In machine learning models, we resize each image into one dimensional. With this dataset, we could do several predictive tasks. For example, we could encode the normal chest X-ray images and detect abnormal chest X-ray images, or we could predict what disease the people might have from chest X-ray images. Since our main goal is to correctly detect different chest diseases, we choose to predict what disease the people might have.

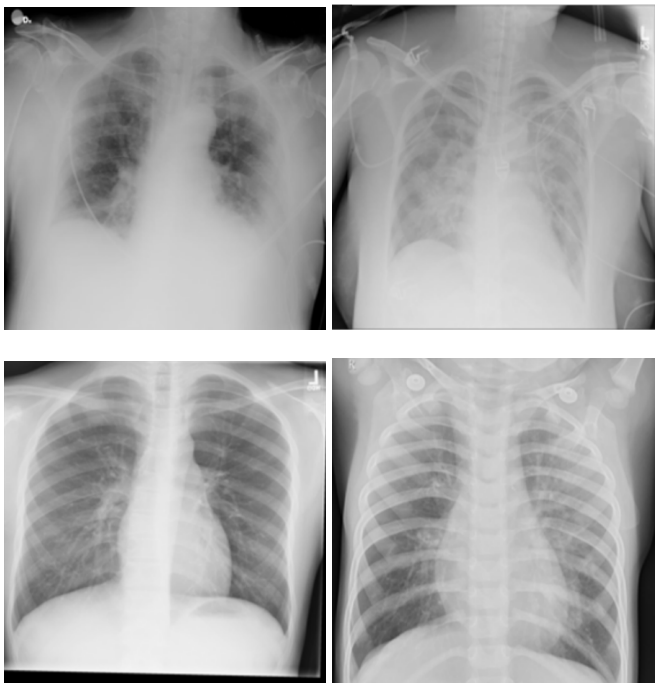

Fig. 1. Different classes of datasets (Top-Left) Covid-19 (Top-Right) Lung Opacity (Down-Left) Normal (Down-Right) Viral Pneumonia

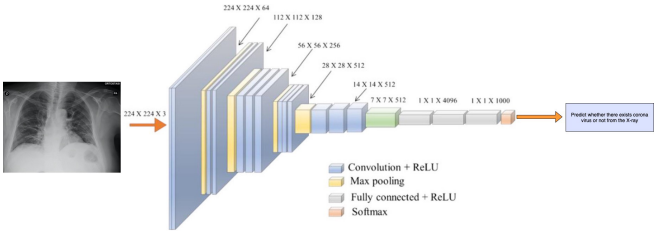

Fig. 2. The structure of VGG19

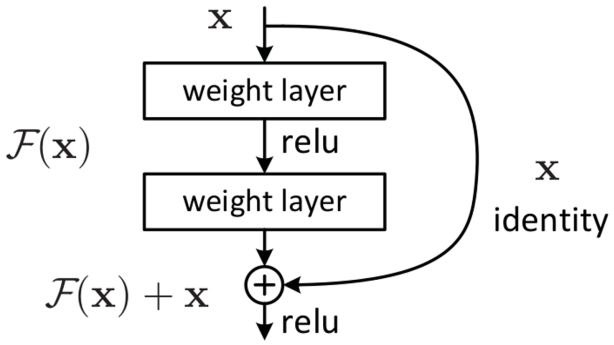

Fig. 3. The structure of ResNet

4. METHOD

We compare the accuracy among different models by using both machine learning and deep learning. In machine learning, we use K-nearest neighbor and Random forest to build up the models. Besides, in deep learning, since CNN models are powerful in classifying images, we select 5 derivative models: VGG19, RestNet, DenseNet, InceptionV3, and Xception. In the following, we briefly introduce these models.

4.1. K-nearest neighbor

K-nearest neighbor is a non-parametric classification method. It is operated in statistical pattern recognition and depends on a training set of pattern vectors among each class. As an unknown vector is inputted, its K closest neighbors are found among these pattern vectors and the class label is decided by the majority[4, 5] In this study, we flatten our image dataset to one dimension, and set K is 8.

4.2. Random forest

Random forest is a tree-based classifier. It relies on growing an ensemble of decision trees and predicts results with the most votes; specifically, we send an image down every tree, average the reached posterior distribution of each feature, and then take the arg max as the classification[6]. In the process, we first resize our image dataset from three dimensions to one dimension and then retain 20% data as the testing part.

4.3. VGG19

VGG19 has 19 layers. It is composed of 5 blocks of convolutional layers; besides, it accompanies pooling layers in the front and fully connected layers at the end. Instead of using 11*11, 7*7, or 5*5 convolutional layers, VGG19 only uses 3*3 convolutional layers shown in Figure 2[7]. It stacks more convolutional networks so the receptive field will be the same in the end. Also, this method reduces the number of parameters in the model and reduces the burden of the memory.

4.4. ResNet

ResNet has many blocks of convolutional layers, but it only has one fully connected layer at the end. Each convolutional layer is used to learn the residual of the network as shown in Figure 3[8]. It uses network layers to fit a residual mapping instead of directly fitting a desired underlying mapping. This shortcut structure can allow the gradient to update the weights in upper layers more easily, which can prevent gradient vanishing in the training process.

4.5. DenseNet121

DenseNet connects each layer to every other layer and acquires additional inputs from all preceding layers; that is, each layer is receiving collective knowledge from all preceding layers. So, the last layer could directly receive the information from the first layer. It can improve the declined accuracy because it could fix the problem of vanishing gradient which happens very often in high-level neural networks. In the various versions of DesenNet, we adopt DesenNet121[9], a classic one, which has fewer layers compared to DesenNet169, DesenNet201, and DesenNet264.

4.6. InceptionV3

InceptionV3 is the third edition of Google's Inception CNN. It automatically converts the image size to 299*299. Compared with the AlexNet, another popular model of CNN, it remarkably reduces the number of network model parameters and asymmetric convolution kernels by increasing the diversity at the same time instead of averaging the pooling. It avoids overfitting by increasing the number of layers to enhance the nonlinear expression of the network. Besides, to allow multiple sizes to operate on the same level in a lower cost, this model also limits the number of input channels by adding an extra 1*1 convolution before the 3*3 and 5*5 convolutions[10].

4.7. Xception

Xception is one of the CNN networks based on separable convolution layers. In this model, the mapping of cross-channel correlations and spatial correlations in the feature maps are completely decoupled. It has 36 convolutional layers. All of them are constructed into 14 modules that have linear residual connections except for the first and last one[11]. This design results can be understood as an Inception module with a maximally large number of towers. Besides, it results in some advantages, for example, it can be easily defined or modified; specifically; it can only take 30 to 40 lines of code to use a high-level library.

5. EXPERIMENT AND RESULTS

Model	(w/o) Fine-Tuning	With Fine-Tuning
VGG19	84.25%	92.25%
ResNet50	69.50%	91.75%
DenseNet121	48.75%	91.25%
InceptionV3	51.00%	89.50%
Xception	63.00%	89.00%

Table 1. Results on transfer learning with different models

Model	Accuracy
k-Nearest Neighbors	73.25%
Random Forest	82%

Table 2. Results on machine learning methods

We conducted the classification in seven different kinds of models with two machine learning methods and five deep learning methods.

In machine learning, we first flatten the image dataset to one dimension and then build the models by using TensorFlow. For K-nearest Neighbor, we allow all points in each neighborhood are weighted equally and decide the most appropriate algorithm to fit the method. For the Random forest, we have 100 trees in the forest. Each node is expanded until all leaves are pure or until all leaves contain less than the minimum sample of the split samples. The maximum feature is the square of the number of features. Besides, we bootstrap the samples as we build tress, and all weights are the same in the process.

For deep learning methods, we try two different kinds of setting to get experiment results. One is using the direct weights pretrained by the Imagenet dataset and replacing the fully connected layers with our own fully connected layer with softmax to get the 4 classes classification accuracy. The other is as same as the previous one, except for adding the fine-tuning step. After training the fully connected layer with the pretrained weights on convolution layers, we set the trainable of the layer to true to unfreeze all the weights in convolution layers. By doing this, the weights pretrained by the Imagenet dataset could fit into the current Covid-19 dataset more. We also set the learning rate in the fine-tuning step really small to let the weights slightly change. In all models, we use the cross-entropy loss as our loss function and Adam as the optimization function with batch size 32, image shape 224*224*3 and 4 classes.

Table 1 shows the results of different models with and without fine-tuning. Among all of the methods, VGG achieves the highest accuracy 92.25% and followed by 91.75%. produced by ResNet50. As for machine learning methods, Random forest achieves the highest accuracy of 82%. The results are shown in table 2.

6. DISCUSSION

Figure 4 shows the the accuracy and the loss versus epochs in VGG19. We only need 20 epochs to see the validation is saturated. Figure 5 also shows that how the loss and accuracy changes in the fine-tuning part. It seems that we only need to use 5 to 10 more epochs to boost up the performance. Compared to similar work last year in ECE228, we beat all of them with about 10% accuracy. We think it results from that

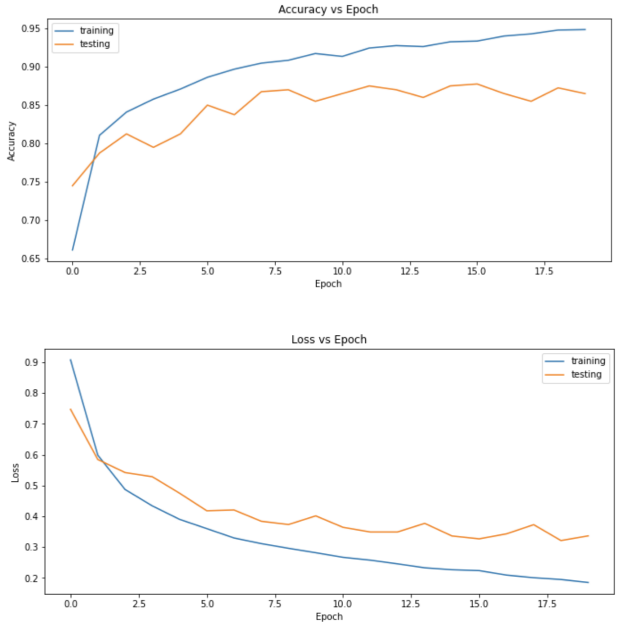

Fig. 4. Accuracy and loss of VGG19 pretrained model in 15 epochs before fine-tuning

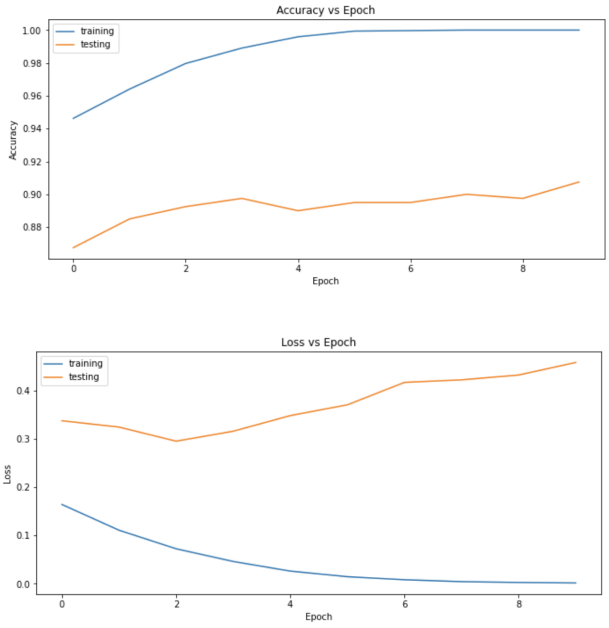

Fig. 5. Accuracy and loss of VGG19 pretrained model in fine-tuning

the dataset is larger currently. We use about 1000 images in each class, so there are 4000 images to train and validate the model, which is much more than last year. And we also use the fine-tuning methods to boost our results, you can see that there is a huge difference with and without fine-tuning. After using the initial weight of VGG from the Imagenet and adding our own fully connected layer to train the model, we unfreeze the whole model with a very low learning rate to fine-tune the model. And fortunately, we got decent results.

7. CONCLUSION

In this study, to classify the Covid-19 images of the dataset, we compare the performance among 7 different models (including machine learning and deep learning). First, we compare the models in deep learning with fine-tuning and without fine-tuning. The results indicate that the models with fine-tuning increase 8-42% accuracy, which implies that performance is better as the model is operated with fine-tuning. It indicates that fine-tuning can improve performance in deep learning. Furthermore, among these models, we find the VGG19 with fine-tuning is the best, in which accuracy is 92.25%.

8. CONTRIBUTIONS

8.1. Chih-Chieh Chien

I took responsibility for the machine learning and part of deep learning (InceptionV3 and Xception). I also wrote method, conclusion, and references.

8.2. Cheng-Yu Chen

I built the skeleton code which preprocesses the data and tried to build custom CNNs. I wrote Abstract, Introduction, Related Work, Dataset and feature in the final report.

8.3. Yun-Yi Lin

I was responsible for the deep learning part of the code and the experiment. I also wrote the experiment and result, and discussion part, and plot the figures in the final report.

9. REFERENCES

[1] Muhammad E. H. Chowdhury, Tawsifur Rahman, Amith Khandakar, Rashid Mazhar, Muhammad Abdul Kadir, Zaid Bin Mahbub, Khandakar Reajul Islam, Muhammad Salman Khan, Atif Iqbal, Nasser Al Emadi, Mamun Bin Ibne Reaz, and Mohammad Tariqul Islam. Can ai help in screening viral and covid-19 pneumonia? *IEEE Access*, 8:132665–132676, 2020.

[2] Antonios Makris, Ioannis Kontopoulos, and Konstantinos Tserpes. Covid-19 detection from chest x-ray images using deep learning and convolutional neural networks. *medRxiv*, 2020.

[3] Hoon Ko, Heewon Chung, Wu Seong Kang, Kyung Won Kim, Youngbin Shin, Seung Ji Kang, Jae Hoon Lee,

Young Jun Kim, Nan Yeol Kim, Hyunseok Jung, and Jinseok Lee. Covid-19 pneumonia diagnosis using a simple 2d deep learning framework with a single chest ct image: Model development and validation. *J Med Internet Res*, 22(6):e19569, Jun 2020.

[4] K. Fukunaga and P.M. Narendra. A branch and bound algorithm for computing k-nearest neighbors. *IEEE Transactions on Computers*, C-24(7):750–753, 1975.

[5] J. Laaksonen and E. Oja. Classification with learning k-nearest neighbors. In *Proceedings of International Conference on Neural Networks (ICNN'96)*, volume 3, pages 1480–1483 vol.3, 1996.

[6] Leo Breiman. Random forests. *Machine learning*, 45(1):5–32, 2001.

[7] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014.

[8] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.

[9] Gao Huang, Zhuang Liu, Laurens Van Der Maaten, and Kilian Q Weinberger. Densely connected convolutional networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4700–4708, 2017.

[10] Christian Szegedy, Vincent Vanhoucke, Sergey Ioffe, Jon Shlens, and Zbigniew Wojna. Rethinking the inception architecture for computer vision. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2818–2826, 2016.

[11] François Chollet. Xception: Deep learning with depthwise separable convolutions. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1251–1258, 2017.