# Transfer learning and Knowledge distillation using Resnet, EfficientNet and NFNet on custom dataset

Bianzhu Fu
bfu38@gatech.edu

Yuqing Ren
yren97@gatech.edu

Chuong Dao
cdao9@gatech.edu

## Abstract

*In this project, we apply transfer learning and knowledge distillation on a custom Covid-19 X-rays dataset. Three different convolutional neural network (CNN) architectures: Resnet [2], NFNet[1], and EfficientNet [8] are applied. The effectiveness of transfer learning with different neural network sizes, neural network architecture, and transfer learning methods are studied and analyzed thoroughly. It is found that simply using CNN as a feature extractor tends to underfit the data and the model has a high bias. In addition, fine-tuning the weights of the CNN tends to overfit the data and introduces high variance and the best model performance needs some architecture modification such as a dropout layer in addition to the pre-trained model. With transfer learning, the highest test accuracy of 0.898 is achieved with Resnet while EfficientNet and NFNet are not too far behind with test accuracies of 0.895 and 0.891 respectively. Knowledge distillation is also applied to understand better how knowledge is transferred between large models to smaller CNN. The effectiveness of different student models and teacher models are studied.*

## 1. Introduction

### 1.1. Background

With the fast spread of Covid-19 worldwide and the near exhaustion of medical resources especially in rural areas, state-of-the-art (SOTA) deep learning techniques in computer vision such as convolutional neural networks (CNN) would show some benefits in rapidly assisting medical diagnosis through medical images. The speed of diagnosis is a critical factor for successful treatment and slowing down the virus widespread. However, there are several challenges in applying CNN for medical image analysis. First, a typical deep CNN needs a large amount of labelled training data to train the model. Due to the cost and privacy requirement, such big data sizes with labels are generally not available for medical images. Secondly, the computational resource budget is often limited, thus, models with too many parameters need to be compressed to fit the limited computation resources. To address these two challenges, transfer learning with knowledge distillation is proposed for the custom Covid-19 X-rays dataset. Transfer learning is a well-known technique for speeding up the medical image diagnosis because it helps overcome the problem of small datasets[10]. Currently, there are two common practices for applying transfer learning with CNN: CNN as a feature extractor and fine-tuning the weights of pre-trained models [10]. However, there are no extensive studies regarding transfer learning method, model size and architecture selection for a given problem among widely available pre-trained models such as Resnet [2], VGG [7], EfficientNet [8], and NFNet [1] et al. This study addresses these practical application questions in transfer learning. We set the goals to study the performances of three different SOTA CNN architectures: Resent, EfficietNet, and NFNet and their variants with respect to transfer learning. Moreover, to transfer the model performance to smaller models, knowledge distillation is also studied with different teacher models and student models. The overall study could potentially provide some guidance on the practical application side of transfer learning and knowledge distillation's model and method selection for medical image analysis.

### 1.2. EfficientNet

EfficientNet is a family of models that achieve state-of-the-art accuracy on ImageNet dataset with up to 10 times better efficiency by relying on fewer parameters. Table-1 lists the weights size of 5 models trained for the same task. EfficientNet-B0, the baseline network found by neural architecture search, has a MobileNetv2 similar architecture. EfficientNet-B1 to B7 are obtained by scaling up the baseline network using a compound scaling method, which uniformly scales up all dimensions, including width of channels, depth, and resolution of input images in a principled way. Research [8] shows that this scaling method consistently improves accuracy and efficiency compared to arbitrary scaling a single dimension. This is justified by the intuition that if the input image is bigger, then the network needs more layers to increase the receptive field and more

channels to capture more fine-grained patterns on the bigger image. EfficientNet performs well on multiple datasets (CIFAR-100, Flowers, etc), making it a good candidate for transfer learning. We only experiment from B0 to B3 due to GPU resource constraints.

### 1.3. NFNet

NFNet (Normalizer-Free Resnet) promotes a new technique called active gradient clipping (AGC) in place of batch normalization and shows that it is able to attain state-of-the-art performance in less training time. Table-2 lists variants of NFNets configurations from DeepMind's paper [1]. These models will be used in the experiment.

### 1.4. XRay Datasets

The X-Ray image data is of size 3.67GB, with 4 classes (Covid-19, Normal, Pneumonia-Bacterial, Pneumonia-Viral), and split into training, validation, and test sets with a ratio of 0.6:0.2:0.2. The data is available and can be downloaded at [6]. tSNE visualization is performed on training data to examine the distribution of classes, shown in Figure-1. Both Covid-19 and Normal classes are well spaced apart, but Bacterial and Viral pneumonia samples intermix with each other, indicating that these 2 diseases are somehow similar, probably in terms of symptoms and causes. Both groups are closer to Normal which makes Covid-19 the most outstanding class. From a classification point of view, this might be a good sign, suggesting Covid-19 could be easier to detect.

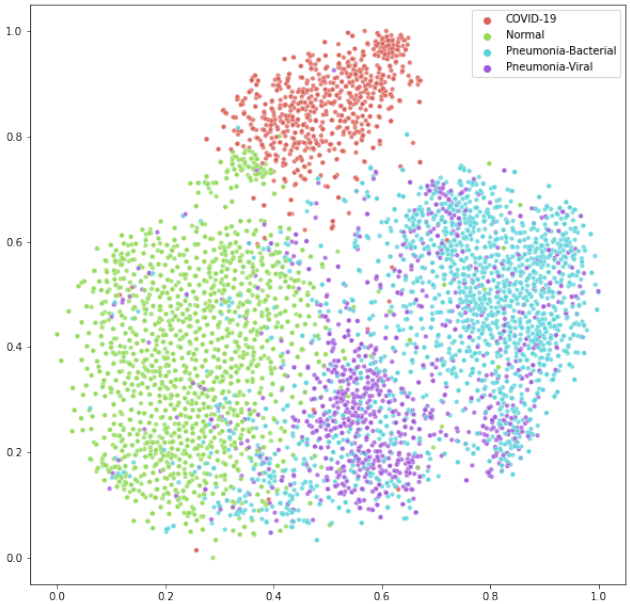



Figure 1. X-Ray Dataset tSNE Visualization

## 2. Approach

### 2.1. Resnet

Different sizes of Resnet are used as feature extractor and fine-tuned network. It is found that if it is used as a feature extractor, the model is limited by capacity and has a high bias. With fine tuning, it over-fits the data and has a high variance. Therefore, 4 dropout layers with dropout size 0.05, 0.05, 0.1, and 0.1 are added to the pre-trained Resnet18 and it improves the result to a test accuracy of 0.898 for resnet18, higher than Resnet34 and Resnet50 as feature extractor and fine-tuned network. For all models, a learning rate of 0.001, weight decay 0.001, momentum 0.9 with SGD optimizer are used. The training episodes for transfer learning (convnet as feature extractor and simply fine-tuned convnet) are 20, and for training from scratch and modified architecture are 40. For all models, early stopping is used and the model with the best validation accuracy is kept for testing.

### 2.2. EfficientNet

For each network (EfficientNet-B0, B1, B2, B3), the results are examined under 3 scenarios when the network is either used as a feature extractor, or fine tuned with or without additional layers. Parameters are tuned individually for each scenario of each network. For example, the learning rate can be set higher for feature extractor than finetuning which has much more parameters to learn. Different final layers are added to each network. Some layers may cause overfitting quickly for B2 but not for B1. One interesting finding is when a model (B1) trained with larger learning rate (0.006) results in overfitting, its test performance is almost always better (88%) than the same model trained with smaller learning rate (0.001) whose learning curve indicates no overfitting. Test accuracy is higher even when validation loss is higher. If the best model is chosen based on the lowest val_loss rather than the highest val_acc, it usually yields poorer test accuracy. This could be related to how the loss is calculated using class probabilities. Sometimes, a dropout ratio of 0.5 might be more susceptible to overfitting than a ratio of 0.3 for the same network (B2) due to randomness. During experiments we use different batch sizes limited by CUDA memory availability, as well as different input image sizes, since the resolution is scaled up for larger models. Due to these reasons, it is necessary to tune the parameters individually for each scenario.

### 2.3. NFNet

The performances of various pre-trained NFNet models (f0 to f3) on classification tasks are studied. In order to simplify the fine tuning procedure, the high-level deep learning library from fast.ai v2 is combined with the PyTorch Image Models (Timm) [9] library to obtain the architecture

| | B0 | B1 | LeNet | ResNet18 | VGG16 |
|---|---|---|---|---|---|
| Weights in MB | 17 | 25 | 27 | 43 | 350 |

Table 1: Comparison of Weight Size between EfficientNet and Other Models

| Variant | Number Blocks for Residual Stages | Number Params | FLOPs |
|---|---|---|---|
| F0 | [1, 2, 6, 3] | 71.5M | 12.38B |
| F1 | [2, 4, 12, 6] | 132.6M | 35.54B |
| F2 | [3, 6, 18, 9] | 193.8M | 62.59B |
| F0 | [7, 14, 42, 21] | 254.9M | 114.76B |

Table 2: NFNet Variants Configurations

and pretrained weights for NFNet variants from F0 to F3. Due to the limitation of Google Colab Hardware memories, a trial of larger models from F4-F6 is not successful. Moreover, the knowledge distillation capabilities of these NFNets variants with a simpler model LeNet as a student are also investigated. This experiment provides some insights to compare with Resnet and EfficientNet as well as obtain some good intuitions if AGC has any negative effects on knowledge distillation of a large model. In addition, all the NFNet variants (F0-F3) are also trained from scratch on the custom Covid-19 dataset in order to further the understanding of the AGC effectiveness and compare its performance with the transfer learning models.

## 3. Experiments and Results

### 3.1. Transfer Learning

#### 3.1.1 Transfer learning with Resnet

The transfer learning results for Resnet is shown in Figure-2(a). There are a few observations: 1) For the 3 different methods: convnet as feature extractor, fine-tuned convnet, and convent trained from scratch, the performance is fairly close for the 3 different model sizes, indicating Resnet18 already has a big enough capacity for this dataset. 2) For the 3 different network sizes: Resnet18, Resnet34, and Resnet50, the performance is the best with fine-tuned network, training from scratch is the second, and Convnet as feature extractor is the worst with the same network size. The learning curve for Resnet18 as a feature extractor is shown in Figure-2(b), implying the performance is limited by model capacity because only the final linear layer can be trained. The learning curve for fine-tuned Resnet18 shows significant over-fitting only after a few training episodes, indicating variance is the major error. Both models are not optimum. Therefore, to improve the performance of the model, 4 dropout layers with dropout size 0.05, 0.05, 0.1, and 0.1 are added to the pre-trained Resnet18 and the learning curve is shown in Figure-2(d). It can be seen that overfitting is improved and an improvement around 2% is observed as compared to simply fine-tuned Resnet18, Resnet34, and Resnet50. A test accuracy of 0.898 is achieved with this architecture modification. One drawback with this modification is that more training episodes are needed, as compared to convnet as feature extractor and simply fine-tuned convent, which only needs smaller than 10 episodes to achieve good performance.

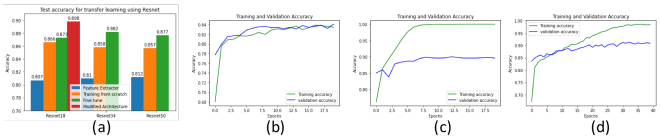



Figure 2. (a) A summary of test accuracy for different Resnet model sizes and structures. (b) learning curve for Resnet18 as a feature extractor. (c) learning curve for fine-tuned Resnet18. (d) learning curve for modified Resnet18 architecture with dropout layers.

#### 3.1.2 Transfer learning with EfficientNet

Our experiment shows Finetune with more layers always performs better than either Finetune alone or Feature Extractor. Larger network if fine tuned, might perform better than smaller network but the difference is not significant. Final results are in Table-3. The best model (0.895) is based on B3, followed by a BatchNorm layer and a dropout layer. Its ROC/AUC, confusion matrix are reported in Figure-3, 4. It is notable that the accuracy of class 3 is consistently lower in all tests. See Figure-5. But our dataset is not extremely imbalanced. The number of samples per class is 768:1962:1800:993 for the training set. I tried to apply focal loss but was only able to increase class 3's accuracy at the cost of lowering other classes' performance. The results from EfficientNet-B0 with CE loss and Focal loss are compared in Table-4.

In fact, viral and bacterial pneumonia are more similar

| Test Accuracy | B0 | B1 | B2 | B3 |
|---|---|---|---|---|
| Feature Extractor | 0.813 | 0.833 | 0.815 | 0.81 |
| Finetune | 0.836 | 0.858 | 0.848 | 0.867 |
| Finetune+MoreLayes | 0.879 | 0.889 | 0.892 | **0.895** |

Table 3: EfficientNet Test Accuracy for Various Methods

| Test Accuracy | Overall | Class 0 | Class 1 | Class 2 | Class 3 |
|---|---|---|---|---|---|
| CE Loss | 0.879 | 0.9728 | 0.9572 | 0.8686 | 0.6717 |
| Focal Loss | 0.8248 | 0.9689 | 0.9526 | **0.6356** | **0.8042** |

Table 4: EfficientNet B0 CE loss vs Focal loss

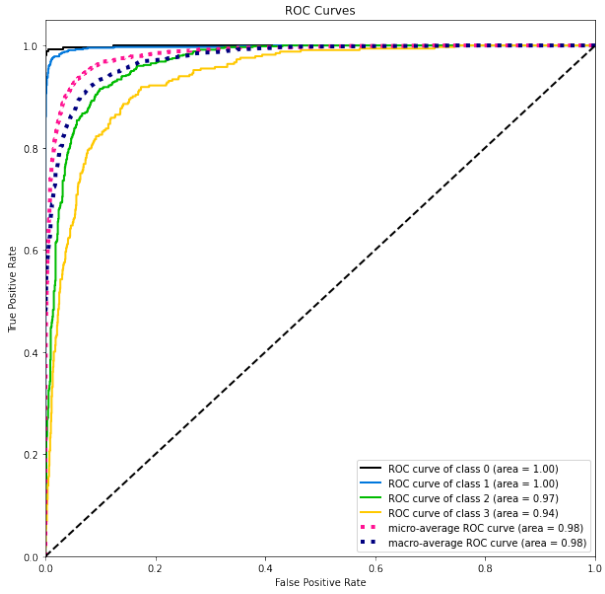



Figure 3. EfficientNet ROC/AUC


Figure 4. EfficientNet Confusion Matrix

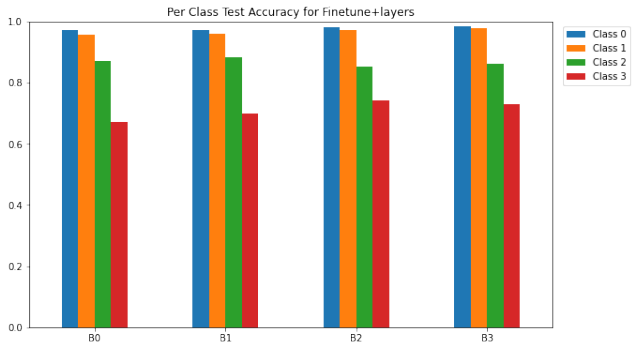



Figure 5. EfficientNet Per Class Accuracy

in nature than Normal or Covid-19. This is shown in the tSNE visualization. According to pharmaceutical company Pfizer, it is challenging but important to be able to differentiate between viral and bacterial pneumonia since bacterial pneumonia can be treated with antibiotics. While radiograph helps, a combination of clinical symptoms, exam findings, and imaging is often needed to uncover the most likely culprit.

We also recognize that it is difficult to achieve overall accuracy beyond 90% after attempting multiple network architectures. We initially applied image augmentation, that is to randomly rotate, scale, shift, and horizontally flip images. In PyTorch, augmentation does not increase the dataset size, but the Dataloader will apply a fresh set of random operations on the fly in any epoch. This turns out to only impact the training accuracy but does not improve the test performance, probably because our test images are not tilted, rotated, or distorted at all. In addition, horizontal flip does not make much sense, because the left lung is always on the right in every image. Maybe a 3:1 split ratio means training data is not enough? We tried to force every training phase to go through 2 epochs before starting the validation phase. This essentially doubled the size of training data, and the new training samples should not be duplicate since

we applied random operations. The training took longer as expected but no obvious improvement was observed. After all, X-ray images are different from natural images, and our pretrained weights are based on ImageNet. This might explain why Feature Extractor's result is always worse than fine tuning.

#### 3.1.3 Transfer Learning with NFNet

For NFNet, after some initial trials, the number of epoch is set to be 15 for the fine tuning process of each NFnet pretrained variant and using the base learning rate 0.003. The batch size is 32 due to GPU's memory limitation for F0-F2 and reduces to 16 for F3. All fine tuning processes happen on Google Colab using 16GB NVidia GPU and using fastai v2 [4] api fine_tune which automatically first freeze the whole pretrained model and trained only the last linear layer to gain some stability weights before unfreeze the whole model and train all layers for 15 epochs. Table-5 shows the test accuracy for each NFNet variants and training time. The test set confusion matrices of each of the NFNet variants in Table-6 in order to compare the performance for each class.

According to the results, it is found that NFNet F2 variant is the best model with test accuracy of 89.15% for NFNet variants and it took about 56 minutes for the fine tuning process to complete. From the confusion matrices, F2 and F3 variants perform slightly better on class Pneumonia-Bacterial and Pneumonia-Viral which account for the higher accuracies. On the other hand, experiments with training NFNet variants F0-F2 from scratch are also performed in order to show transfer learning is a more superior approach for custom datasets. We train each variant from scratch for 35 epochs and recording the result in the Table-7. As a result, transfer learning proves to be better at both training time and test accuracy.

#### 3.1.4 Transfer Learning Summary

From all 3 network architectures, the following conclusions can be drawn: 1) the performance is best with fine-tuned network architecture with a pre-trained model. Transfer learning with CNN as a feature extractor performs the worst. 2) For the same class of network type, a bigger model size does not necessarily yield better performance with transfer learning for both CNN as a feature extractor and fine-tuned CNN. 3) The best test performance with Resnet, EfficientNet, and NFnet are: 0.898, 0.895, and 0.892, respectively. This indicates the model architecture does not affect the transfer learning as much for this data. The performance is probably more limited by the data difference between the pre-trained model (ImageNet) and X-rays images [5].

### 3.2. Knowledge Distillation

Knowledge distillation is a technique to transfer knowledge from a trained teacher network to another student network to assist the training process of the student network. The key for knowledge distillation is the "dark knowledge" from soft labels, which is controlled by the temperature parameter in the modified SoftMax function [3]. The dark knowledge helps the student model to form a decision boundary and/or probability distribution similar to the teacher's model. The joint loss function and knowledge distillation is implemented from scratch according to [3]. For resnet, the teacher is a trained modified Resnet18 with dropout layer, with a test accuracy of 0.898 being achieved. The results with and without knowledge distillation for different student models are presented in Table-8. Here are a few interesting observations: 1) When the student model has a high capacity (Resnet34 and Resnet50 for Resnet), knowledge distillation is not very useful, and improvement is negligible. This might be due to the reason that the decision boundary for the student model is more complex than the teacher, thus, it is difficult to enforce the teacher's boundary to the student's decision boundary. 2) With very small capacity (frozen resnet18 with one linear layer), knowledge distillation is not effective either due to the student's low model capacity, failing to follow the probability distribution of the teacher's prediction. With improved model capacity (frozen Resnet18 plus 2 fully connected layers), the improvement with knowledge distillation is still negligible (+0.004). This might be due to the fact that with all convolution layers frozen in Resnet18, even with a higher model capacity, the architecture of the model makes the knowledge distillation ineffective, since the decision boundary shape is formed by convolution layer as a kernel before feeding into the final fully connected linear layer. Therefore, an addition of a fully connected layer in the end is not likely helping shape the decision boundary. This is further confirmed by the LeNet student model, which shows an improvement of 3.8 percent with knowledge distillation with a test accuracy of 0.857, similar to the accuracy of Resnet18, Resent34, and Resnet50 trained from scratch.

For EfficientNet, the distillation loss was implemented using KL divergence rather than the cross entropy with soft targets from student and teacher as suggested in [3]. We chose the model with test accuracy 0.895 as the teacher. Students include LeNet, Feature Extractor and Fine tuned B0 to B3. To ensure results are comparable, we use the same parameters for all experiments: SGD learning rate=0.001, no scheduler, epochs=10, patience=3, temperature=3, alpha=0.1. The most significant boost occurs on the simplest network LeNet with 4.77% improvement. For B0 to B3, Fine tuned models are also improved, even B3 was seeing 2.77% increase within 10 epochs which is impressive

| Variant | Training Accuracy(%) | Test Accuracy(%) | Training Time(minutes) |
|---|---|---|---|
| F0 | 90.0 | 87.96 | 30.25 |
| F1 | 90.9 | 87.90 | 41.25 |
| F2 | 90.22 | 89.15 | 56.25 |
| F3 | 89.95 | 88.72 | 90 |

Table 5: NFNet Transfer Learning Results

considering the teacher is also based on B3. All Feature extractors show slightly poorer results. This is understandable because all layers' weights except the last layer are frozen. The model's learning capacity is too limited to absorb the dark knowledge distilled from the teacher. The result in Table-9 shows that knowledge distillation is effective.

We also conduct another knowledge distillation experiment by training the same student Lenet Model with variants of NFNet. The result in Table-10 shows NFNet F2 is the best teacher model that inspires the student's accuracy at 0.839. It is interesting to observe that the student model accuracies went down approximately 2% with F3 as teacher. We suspect the gap between the model sizes between NFNet F3 and Lenet is having some effects on the ability to learn of the student model. On the other hand, the accuracy of student models with F0, F2 as teachers only differed by 0.5% but the training time is about 70 minutes longer for F2. Another interesting point of the experiment is that the size of the trained Lenet student model is only 15MB while the size of the NFNet F0, NFNet F1, NFNet F2 are 204MB, 507MB, and 740MB respectively. It could be an interesting trade-off between accuracy vs inference speed and model's size if the classification task does not require higher accuracies than 81%.

## 4. Discussion

We reinforced our understanding on transfer learning as well as being able to dive deep into the network architectures of some of the trending state-of-the-art models such as NFNet and EfficientNet. We are able to directly apply our newly acquired deep learning techniques from the course such as Focal Loss, tSNE, Knowledge Distillation to the project. Our transfer learning results have shown the great advantages of quick training time and high accuracy of pretrained models in custom classification tasks. Each team member achieved at least 89% test accuracy using different networks and approaches. Our implementation on knowledge distillation is effective. Therefore, we consider this project a success.

## 5. Work Division

The work load distribution for each team member is shown in Figure-6. All students equally contributed to final report editing and compiling, multiple team meetings per week, code reviews and paper reviews.

| Student | Contributed Aspects |
|---|---|
| Bianzhu Fu | ResNet evaluation and Knowledge Distillation analysis |
| Chuong T Dao | NFNet evaluation and Knowledge Distillation analysis |
| Yuqing Ren | EfficientNet evaluation and Knowledge Distillation analysis |

Figure 6. Work load distribution

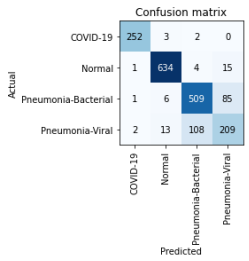
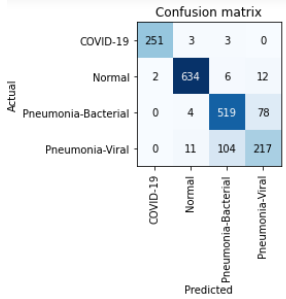
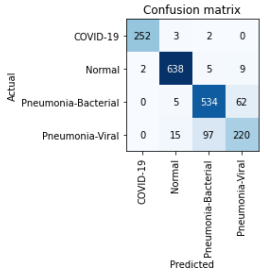
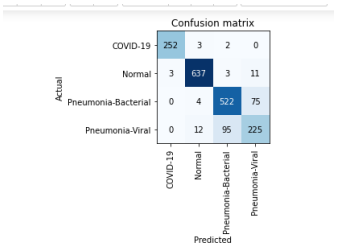
| F0 | F1 | F2 | F3 |
|---|---|---|---|
|  |  |  |  |









Table 6: NFNet Transfer Learning Confusion Matrices

| Variant | Training Accuracy(%) | Test Accuracy(%) | Training Time(minutes) |
|---|---|---|---|
| F0 | 82.20 | 79.50 | 62.3 |
| F1 | 82.78 | 79.22 | 80 |
| F2 | 82.56 | 78.8 | 133 |

Table 7: NFNet Training From Scratch Results

| Student Model | Resnet50 | Resnet34 | Resnet18 | Frozen Resnet18 | Frozen Resnet18 + 2 Fully Connected Layer | Lenet |
|---|---|---|---|---|---|---|
| Scratch | 0.857 | 0.858 | 0.866 | 0.807 | 0.807 | 0.819 |
| KD | 0.858 | 0.872 | 0.871 | 0.805 | 0.811 | 0.857 |
| Improvement | 0.001 | 0.014 | 0.005 | -0.02 | 0.004 | **0.038** |

Table 8: Knowledge Distillation with ResNet

| | | Feature Extractor | | | | Fine Tune | | | |
|---|---|---|---|---|---|---|---|---|---|
| Student | LeNet | B0 | B1 | B2 | B3 | B0 | B1 | B2 | B3 |
| Without KD | 0.8004 | 0.8026 | 0.8150 | 0.8063 | 0.8020 | 0.8562 | 0.8584 | 0.8557 | 0.8519 |
| With KD | 0.8482 | 0.8010 | 0.8080 | 0.7999 | 0.7988 | 0.8774 | 0.8812 | 0.8628 | 0.8796 |
| Delta | **0.0477** | -0.0016 | -0.0070 | -0.0065 | -0.0033 | 0.0211 | 0.0228 | 0.0070 | **0.0277** |

Table 9: Knowledge Distillation with EfficientNet

| Teacher | Student LeNet Train Accuracy(%) | Student LeNet Test Accuracy(%) | Training Time(minutes) |
|---|---|---|---|
| F0 | 84.13 | 80.85 | 56 |
| F1 | 82.94 | 81.29 | 91 |
| F2 | 83.43 | 81.39 | 126 |
| F3 | 82.50 | 79.44 | 192 |

Table 10: NFNet Knowledege Distillation Results

## References


[1] Andrew Brock, Soham De, Samuel L. Smith, and Karen Simonyan. High-performance large-scale image recognition without normalization. *CoRR*, abs/2102.06171, 2021. 1, 2

[2] 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 (CVPR)*, June 2016. 1

[3] Geoffrey Hinton, Oriol Vinyals, and Jeff Dean. Distilling the knowledge in a neural network, 2015. 5

[4] Jeremy Howard and Sylvain Gugger. fastai: A layered API for deep learning. *CoRR*, abs/2002.04688, 2020. 5

[5] Maithra Raghu, Chiyuan Zhang, Jon M. Kleinberg, and Samy Bengio. Transfusion: Understanding transfer learning with applications to medical imaging. *CoRR*, abs/1902.07208, 2019. 5

[6] Gokul Lal; Prajapati Sunny; Bhaumik Rahul; Kumar Tarun; S Sanjana; Bhalla Kriti (2020) SAIT, UNAIS; k v. Curated dataset for covid-19 posterior-anterior chest radiography images (x-rays), 2020. 2

[7] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. In Yoshua Bengio and Yann LeCun, editors, *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*, 2015. 1

[8] Mingxing Tan and Quoc V. Le. Efficientnet: Rethinking model scaling for convolutional neural networks. *CoRR*, abs/1905.11946, 2019. 1

[9] Ross Wightman. Pytorch image models. `https://github.com/rwightman/pytorch-image-models`, 2019. 2

[10] Fuzhen Zhuang, Zhiyuan Qi, Keyu Duan, Dongbo Xi, Yongchun Zhu, Hengshu Zhu, Hui Xiong, and Qing He. A comprehensive survey on transfer learning. *CoRR*, abs/1911.02685, 2019. 1