Dalhousie University - CSCI 4190

Pneumonia Detection on X-Ray Data using Transfer Learning and Augmentation

By

Bruce MacDonald

B00580971

br869474@dal.ca

Seth Piercey

B00725633

piercey@dal.ca

For Dr. S. Oore

August 14, 2018

1 Abstract

The goal of this project is the accurate binary classification of chest x-ray images into classes “yes pneumonia” or “no pneumonia”. However, due to limited time and computing resources, we use models pretrained on ImageNet for this task. To conform these pretrained models to the task at hand, we start by training another, separate layer with the purpose of using the pretrained models as feature extractors for the x-ray dataset. With this base technique, called transfer learning, we then explore removing further layers of each model, preprocessing and augmenting the data, and combining the models into a single collection called an ensemble. Although more work is needed, we achieve results which are competitive with state-of-the-art classifiers in this area.

2 Introduction

The goal of this project is to classify the presence of pneumonia in a patient using their chest x-ray. There are 3 different examples in the dataset: normal, bacterial pneumonia, and viral pneumonia. Our initial goal is the binary classification of whether a patient has pneumonia or not, and we research a variety of models to see which is the most effective in this pursuit. We later show positive results with our methods, approaching the competitive classification rates for this data set with minimal training and resource use.

This is an important problem as an accurate classifier can help physicians in their diagnosis of a potentially life threatening condition. The classifier could also help make physicians aware of possible pneumonia symptoms in the case that the physician is not aware of the possibility.

3 Related Work

There is extensive work related to classifying x-rays, but we mostly focus on publications related to chest x-rays as all our data was x-rays of the chest. The most useful publication was the “Comparison of Deep Learning Approaches for Multi-Label Chest X-Ray Classification” [1].

This paper covers transfer learning using ResNet-50 with and without fine tuning along with training a network from scratch. This prior work was useful in learning about the application of transfer learning related to any domain. This method was more broad than our application, classifying many other conditions as well as pneumonia. Additionally, “Deep Learning for abnormality detection in Chest X-Ray images” by Tataru, Yi, Shenoyas, and Ma, explores classification of chest x-rays by comparing the accuracy of different models and by using data augmentation, which is an approach very similar to ours.

4 Data

A dataset of medical images called “Chest X-Ray Images (Pneumonia)” [2], originally by Kermany et al. as part of a larger dataset but discovered by us on Kaggle, contains 5683 labelled x-rays of the types “normal”, “bacterial pneumonia”, and “viral pneumonia”. Despite the three classifications of pneumonia manifestation, we solely focus on the binary classification of the presence of pneumonia. The total size of all the images is 1.2 GB, and images were pre-split into training and testing data.

As seen in Figure 1, most x-ray images can be identified as positive or negative cases of pneumonia by inspection: images with pneumonia have more of a white haze in the chest/lungs region.

While no processing of the data took place to make the data more usable, some data processing later took place to improve classification results, such as processing to account for biases due to the imbalance of “yes pneumonia” and “no pneumonia” data points; the data contains approximately 4000 “yes pneumonia” examples but only 1500 “no pneumonia” examples. We explore this further in section 6.2.

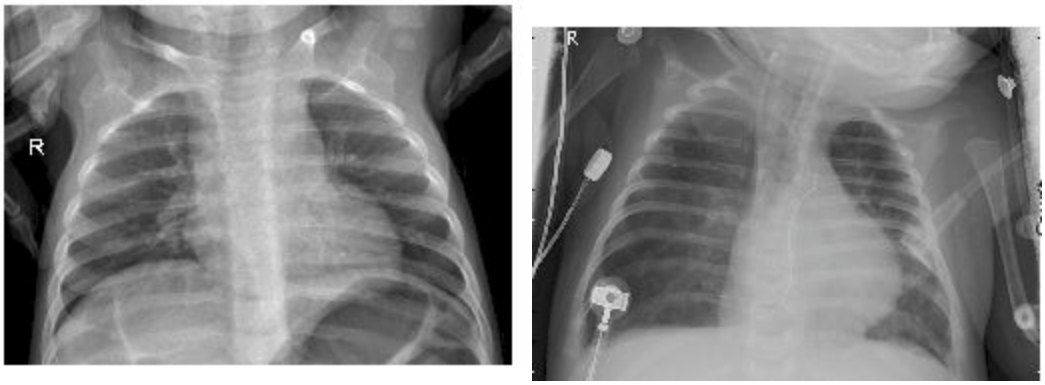

Figure 1: A comparison of no pneumonia (left) and yes pneumonia (right) images.

5 Methods

We approach this problem in two major ways. First, we focus on repurposing the abilities of models trained on other datasets for our purposes in a technique known as transfer learning. This allows us to avoid the laborious process of training our own model "from scratch", since even training a simple model on this amount of data would take too long on what resources are available to us. Additionally, pretrained models have the advantage of being trained with large amounts of data, being simple to use in transfer learning, and having been designed by professionals through years of research. The disadvantage of pretrained models is that are not trained for our specific use case of pneumonia classification. However, in our opinion, the advantages of starting with a pretrained model far outweigh the disadvantages in our case.

Transfer learning works by removing the top-most layer from the pretrained network and retraining that layer to work on the given data. Hence, this technique essentially relegates "feature extraction" of the unrelated dataset to the original network and trains another classifier to work with these extracted features [17]. Going further with this idea, we can also remove more layers than just the top-most and instead replace those layers with ones of our choosing. When we then retrain these layers with our dataset, we get a much more robust classifier for our data. This technique is expanded upon in sections 6.1 and 6.3.

The decision to follow the transfer learning route was aided by surveying code already available on Kaggle created by other users for the same task. By doing so, a major benefit of using Kaggle as a discovery and networking platform, we found the kernel by Paul Mooney [4], which provided us a base to start working with the pneumonia data.

Our second approach to this problem was to transform our data into a set which could be more easily classified. We recognized immediately the need to account for the imbalance of positive and negative cases in the data, which contributes to classifier bias during training. Along with accounting for this imbalance, we also worked on preprocessing the data and augmenting the data with other data points, with mixed results. This approach was combined with both our first attempts and later attempts at transfer learning; this is expanded more in sections 6.2, 6.4, and 6.5.

6 Experiments

6.1 Comparing Pretrained Models

From models available in the Keras library [3], our chosen software for this project, we selected ResNet-50, InceptionV3, and VGG16 as potential candidates for our pretrained network. While these models each take different approaches to classification, they each use the same basic building blocks of convolutions, pooling, dropouts, and fully connected layers. Additionally, they are all convolutional neural networks (CNNs), which is the current standard for image recognition. CNNs are effective at classifying images as their convolutions and filters create feature maps that identify important features of an input image. With this selection of pretrained models, we first try transfer learning by adding and training a single node at the end of the network which will be trained to detect if the output is in the “yes pneumonia” or “no pneumonia” category. This node will be a simple softmax or sigmoid function. Model details are as follows.

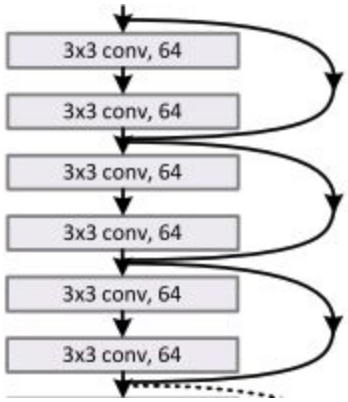

Figure 2: Layers in a residual network [6].

First, we used ResNet-50, a pretrained 50 layer residual network trained on ImageNet (Figure 2). Residual networks have the advantage of shortcut connections which makes learning features from the input of another layer possible. The reason we chose ResNet-50 as an experimental model was due to its use in previous work as a powerful computer vision model [1].

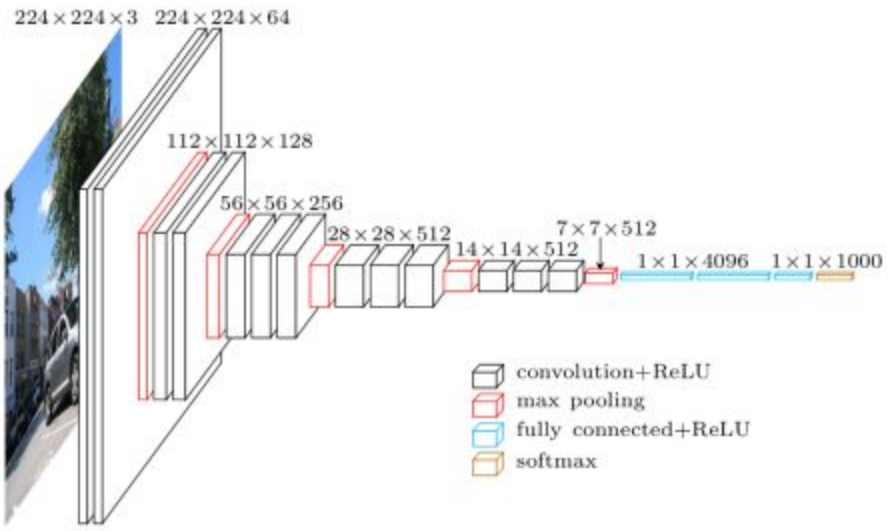

Figure 3: The VGG16 architecture [7].

Next, we used VGG16, a pretrained CNN model (Figure 3). It has less layers than other models and has been shown to generalize very well to many datasets [8].

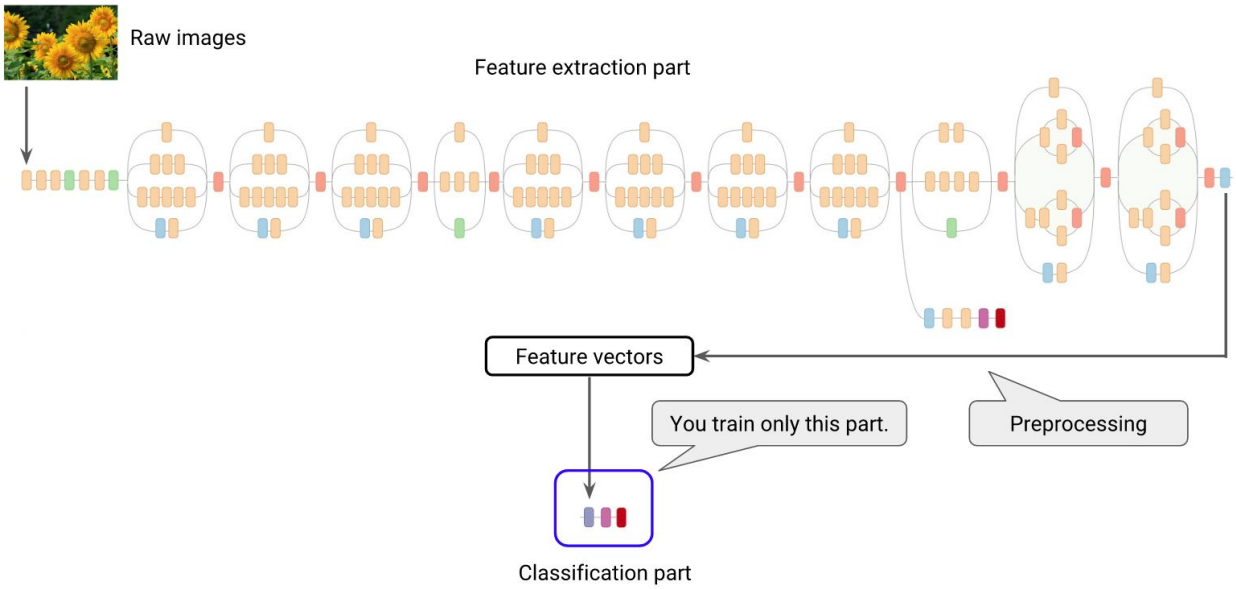

Figure 4: The architecture of InceptionV3 [9].

Finally, InceptionV3 was our last candidate pretrained network (Figure 4). InceptionV3 is a deep network that computes multiple levels of features in each module of the network [10]. It has also shown to be an effective method of detecting abnormalities in x-rays by some research [11].

Model	**Test Loss**	**Test Accuracy**
VGG16	0.455	0.809
InceptionV3	0.792	0.709
ResNet50	2.503	0.625

Table 1: Transfer learning results with a trained *softmax* output node (10 epochs).

Model	**Test Loss**	**Test Accuracy**
VGG16	0.539	0.855
InceptionV3	3.74	0.625

ResNet50	0.673	0.625

Table 2: Transfer learning results with a trained *sigmoid* output node (15 epochs).

Tables 1 and 2 show the results from using the above models to classify the pneumonia dataset with either a sigmoid or softmax output node. Clearly, is the prime candidate for further study on this dataset. Interestingly, loss would continue to fall with more training iterations on InceptionV3 and ResNet50, but the test accuracy improvements would be minimal or reduce. This is probably due to the fact that InceptionV3 and ResNet50 are quite deep neural networks which do not generalize as well as VGG16, a shallower network.

Surprisingly, there was not a major advantage between a sigmoid and softmax output node on VGG16. Sigmoid classifiers are more commonly used in binary yes/no classifiers while a softmax is normally used for classifiers with multiple possible outputs. The sigmoid output has a slightly better test accuracy which makes sense given that our output value should range between 0 and 1.

6.2 Undersampling

In the sample notebook we used as a base for this project [1], a strategy used for dealing with unbalanced data was undersampling. This means removing some of the “yes pneumonia” examples in order to have a 1:1 yes/no example ratio. This is the simplest method of data balancing, but may not be preferable in this case as it means much less training/test data.

Model	**Test Loss**	**Test Accuracy**
VGG16	0.485	0.863
InceptionV3	2.66	0.532

Table 3: Undersampling results with a trained *sigmoid* output node (15 epochs).

In the case of VGG16, having a more balanced set of training/test data helps marginally improve test accuracy. Having less data also helps speed up training which is valuable in our case of limited resources.

6.3 Stripping and Specializing Layers

Once we have determined some viable models through testing them with one unit on the end, we proceed by removing layers off the pretrained models and replacing them with our own. Stripping layers off a model and replacing them with new trainable layers is a generally effective method because the first few layers of a model are the ones that detect general features. We will try different promising pretrained networks again with different amounts of layers removed and replaced with simple depth wise separable convolutions similar to those used in Xception [5].

After some research, another notebook that demonstrated using Xception style convolutions can be found in [12]. Depth wise separable convolutions, as in Xception, are a good choice for our scenario as they have less parameters to train than a typical convolution. The depth wise convolutions will be used in conjunction with batch normalization to reduce covariant shift and ensure even faster training.

Model	**Test Loss**	**Test Accuracy**
VGG16 (raw data)	0.429	0.893
VGG16 (undersampling)	0.491	0.891

Table 4: Results with one layer stripped a trained *sigmoid* output node (15 epochs).

Model	**Test Loss**	**Test Accuracy**
VGG16 (raw data)	0.539	0.917
VGG16 (undersampling)	0.434	0.925

Table 5: Results with two layers stripped a trained *sigmoid* output node (15 epochs).

Tables 4 and 5 show the results of stripping layers from the pretrained VGG16 model. The results for test accuracy and loss saw some minor improvements over relying solely on the output from the pretrained model. Training accuracy and test accuracy followed similar trends in results, with test accuracy having much larger fluctuations (graphs can be seen in accompanying files). Test accuracy was also noticeably higher than training accuracy in some cases, this indicates that the smaller sample of test data may be getting lucky in the random samples being classified correctly.

6.4 Preprocessing

With the difference between some 'yes/no' pneumonia examples being very subtle, we explored some preprocessing techniques that could be used to create a clear difference between the two classes. In order to make the pneumonia, which manifests itself as a sort of "fog" in the x-rays, stand out more, we tried sharpening all the input images. This ended up causing accuracy to become much worse, which is possibly due to the fact that sharpening causes certain insignificant features of the input to unintentionally dominate the classification. This experiment was soon abandoned so that we could turn our focus on other things.

6.5 Data Augmentation

Our next step was to copy some of the dataset images, alter some features of the copy, and reintroduce the copy into the dataset in a process known as data augmentation. For this process, we used the built in ImageDataGenerator Keras function. Augmentations were chosen for their representation of the kinds of transformations a human expert would perform on a physical x-ray to understand it better: small rotations, minor random horizontal/vertical rotations, random zoom, and horizontal flips.

Model	**Model Modification**	**Train Loss**	**Train Accuracy**	**Test Loss**	**Test Accuracy**
VGG16	Sigmoid output	0.119	0.957	0.227	0.922
VGG16	Sigmoid output and last layer	0.166	0.979	0.361	0.954

	replaced with separable convolutions.				
VGG16	Sigmoid output and last two layers replaced with separable convolutions.	0.191	0.974	0.335	0.950

Table 6: Results of training on augmented data and testing with unmodified data (50 epochs).

This augmented data was used with each previous model experiments for training, and achieved the best results in each case. Augmentation showed many advantages over previous methods, a major improvement being in generalization. Without augmentation, loss and accuracy on test data would often decrease as the model became more accurate on the training data, but the opposite happens with augmented training data: as training accuracy increases, test accuracy follows the same trend.

Another improvement augmentation provided was reproducibility. With no augmentation, training would not produce consistent results on different training runs.

6.6 Ensembles

Given the relatively accurate output of our chosen pretrained models, we then implemented an ensemble of VGG16 and Inception. Ensembles are models which are the amalgamation of many different models into one, the idea being that each model can account for the failures of the others. This can be implemented in many different ways, but we chose to implement it by averaging the outputs of Inception and VGG16, which is the simplest method of ensembling. Unfortunately, accuracy was about 75% on the test set -- a reduction from our previous experiments with VGG16, which may be due to the biases of Inception. Still, this is better than random guessing, and so shows promise.

7 Future Work

Any future work in this project should focus on attempting to reduce the amount of false negatives present in the classifications. In other words, due to the medical nature of this project, reduction of the amount of data instances classified as not having pneumonia, but actually do, is crucial for the well-being of patients whose x-rays are processed with a system such as this. This reduction could be achieved through a loss function which greatly penalizes false negatives, which may increase the number of false positives on the testing set, but is regardless much more desirable in a medical setting.

Additionally, it would likely be useful to pursue further work in the use of ensembles on this dataset. As in section 6.6, a first attempt at creating an ensemble with our chosen pretrained models was done, but this could be improved upon. Stacking ensembles, which is when a group of models are trained on the same set of data, and then another model is trained with the output of those models, is a popular ensembling technique for Kaggle competition, and should be explored in this case.

8 Conclusions

Pneumonia is an awful disease of the lungs which is a leading cause of death worldwide [16]. Here we have shown that a system for the accurate detection of pneumonia can be developed, using transfer learning and data augmentation, which will allow for the easy detection of pneumonia in patients using their chest x-rays. With further refinement, this system may be used by medical professionals to reduce the amount of workload they feel for identifying pneumonia in patients; by feeding chest x-rays into a system like the one presented here, medical professionals will only need to review outputted images which may be false negatives -- a small percentage of the whole.

References

[1] Baltruschat, I.M., Nickisch, H., Grass, M., Knopp, T., Saalbach, A.: Comparison of Deep Learning Approaches for Multi-Label Chest X-Ray Classification. ArXiv e-prints. March 2018

[2] Kermany, D.; Zhang, K.; Goldbaum, M. 2018, "Labeled Optical Coherence Tomography (OCT) and Chest X-Ray Images for Classification", Mendeley Data, v2. doi:10.17632/rscbjbr9sj.2

[3] Chollet, F., *Keras.* 2015. https://keras.io/

[4] Mooney P., *Detecting Pneumonia in X-Ray Images:* Kaggle, 2018. Available: https://www.kaggle.com/paultimothymooney/detecting-pneumonia-in-x-ray-images

[5] Chollet, F., Xception: Deep learning with depthwise separable convolutions. arXiv preprint arXiv:1610.02357v2, 2016.

[6] He, K., Zhang, X., Ren, S., and Sun, J. Deep residual learning for image recognition. In IEEE Conference on Computer Vision and Pattern Recognition (2015).

[7] D. Frossard. "Macroarchitecture of VGG16", [Diagram], 2016. Available: https://www.cs.toronto.edu/~frossard/post/vgg16/vgg16.png

[8] Simonyan, K., & Zisserman, A. 2014. Very deep convolutional networks for large-scale image recognition. CoRR, abs/1409.1556.

[9] "Image Classification Transfer Learning with Inception v3", [Diagram]. Available: https://codelabs.developers.google.com/codelabs/cpb102-txf-learning

[10] Rosebrock, A. "ImageNet: VGG16, ResNet, Inception, and Xception with Keras", 2017. Available: https://www.pyimagesearch.com/2017/03/20/imagenet-vggnet-resnet-inception-xception-keras/

[11] Tataru, C., Yi, D., Shenoyas, A., Ma, A. "Deep Learning for abnormality detection in Chest X-Ray images", 2017. Available: http://cs231n.stanford.edu/reports/2017/pdfs/527.pdf

[12] NAIN, *Beating everything with Depthwise Convolution:* Kaggle, 2018. Available: https://www.kaggle.com/aakashnain/beating-everything-with-depthwise-convolution

[14] Szegedy, C., Vanhoucke, V., Ioffe, S., Shlens, J., Wojna, Z.: Rethinking the inception architecture for computer vision. In: CVPR. (2016)

[15] George, Ronald B. (2005). *Chest medicine : essentials of pulmonary and critical care medicine* (5th ed.). Philadelphia, PA: Lippincott Williams & Wilkins. p. 353. ISBN 9780781752732.

[16] "CS231n Convolutional Neural Networks for Visual Recognition", 2018. [Online]. Available: http://cs231n.github.io/transfer-learning/