CP468 - Viral Pneumonia Identification

August 2nd 2024

Dr. Emad Mohammed

Group-8

Michael D’Onofrio, 169032946

Ryan Chisholm, 169027577

Shaun Bangalore, 210912000

Suryanshu Singh, 169031137

Abstract

Pneumonia is a common yet potentially deadly lung infection which afflicts several people around the world. This project aims to use AI image detection to be able to quickly identify if a patient has pneumonia based on an X-ray image of their chest. The current methods to diagnose Pneumonia have long turnaround times which this project aims to shorten. Similar AI imaging models are also being developed for other parts of the medical field, showing that image detection has a very high potential to be integrated into modern medicinal practices.

A dataset of 3500 X-ray images containing healthy and infected lungs were used to train and test our models. The three pre-trained models we used utilized VGG16, ResNet and DesNet architectures. The model we build from scratch uses a CNN approach. Our CNN model has a strong training accuracy of nearly 100%; however, the model has a much lower validation accuracy, suggesting a potential overfitting problem with our model. The VGG16, ResNet, and DesNet models each perform much more accurately when compared to our CNN model, suggesting they experience overfitting to a much lesser degree.

The pre-trained models show impressive results in identifying pneumonia from X-ray images; this shows that the integration of machine learning models have the potential to be a beneficial addition to the process of diagnosing pneumonia.

Introduction to the Project and Discussion of Similar Solutions

Technology has played a crucial part in the development of medicine throughout human history. Ranging from physical to digital, technology has advanced alongside medicine to help detect, treat, and cure deadly illnesses which used to plague humanity. One of humanity's oldest illnesses is pneumonia and "despite improvements in care over the last few decades, pneumonia still causes significant morbidity and mortality worldwide" (*Pates et al., 2023*). Once symptoms become present in a patient, the quick detection and verification of pneumonia is key to a patient's survival. This project aims to make use of machine learning to quickly detect pneumonia from an X-ray in order to cut down on the time it takes for doctors to differentiate if a patient has pneumonia or a different lung affliction.

The current methods of diagnosing pneumonia include Pulmonary Chain Reaction also known as PCR, Antigen tests, Blood Cultures, and X-ray scans. These tests are used in order to identify if a patient has pneumonia and what bacteria may be causing it. The most prevalent downside to these current methods are the time it takes to identify pneumonia and the potential for a misdiagnosis of a different bacterial infection. Microbiological tests such as PCR, Blood Cultures, and Antigen tests can take upwards of one to three days in order to identify the presence of pneumonia and the bacteria causing it (*Heitz et al., 2023*). Although these methods are compound tests, waiting up to three days to confirm a patient has pneumonia may cause doctors to not give a patient the correct treatment within that time. As well as this, microbiological tests also have the chance of returning a misdiagnosis depending on the quality of the sample tested (*Cilloniz et al., 2016*). Traditional X-ray imaging is used to detect the presence of pneumonia in a patient in order to give doctors a clearer idea of how to treat the patient until further tests can be run to detect the bacteria afflicting them. The turnaround time of manually identifying pneumonia by a radiologist can range from a few minutes to an hour depending on the experience of the radiologist. This method takes time and resources away from other patients while the radiologist manually reviews an X-ray. The implementation of AI into automatically identifying pneumonia in an X-ray image could give a near-instant turnaround time, allowing for doctors to be able to prescribe appropriate initial treatments to a patient and allowing radiologists to focus their time elsewhere.

Artificial intelligence is currently being integrated into ophthalmology, dermatology, and oral medicine. The use of AI imaging in the "automated recognition and classification of lesions from clinic images" (*Gomes et al., 2023*) is making headway as some AI trials have proven comparable to human detection. Although these AI models are still being tested, they prove that similar AI imaging to this project is in development for other parts of the medical field. The usage of AI imaging to promote fast and accurate diagnoses is not new to this project which is why this project has the potential to be integrated into the medical industry, similar to how these other models are being tested in their respective fields.

Description of Models and Methodology

For this project, we utilized a lung disease dataset containing around 3500 x-ray images labeled with the presence or absence of pneumonia, available on Kaggle (*Fatemeh, 2024*). This dataset was divided into three separate directories. One for lung opacity, normal results, and those identified with pneumonia in order to ensure robust evaluation of our models. Each directory included x-ray images labeled with either the presence or absence of pneumonia, and in order to ensure diversity within our dataset, was captured in a variety of medical settings. The training set comprises the majority of our images, and allows our model to learn a wide range of features associated with both healthy and pneumonia-affected lungs. The validation set is used during the training process at the end of our project to monitor the model's performance and tune hyperparameters, making sure our model does not overfit to the training data in addition with other techniques. Finally, the testing dataset, which the model has never seen during training, is used to evaluate the model's performance in a real-world scenario.

We implemented two approaches, the first being a CNN model that was built from scratch, as well as a transfer learning model that utilized the pre-trained VGG16 architecture. We also used ResNet and DenseNet for our other two models. VGG16, developed by the Visual Geometry Group at the University of Oxford, is known for its effectiveness in image classification tasks, making it the best choice for our purposes (*Zisserman, 2015*). The CNN model was designed with the typical architecture you would expect, where the convolutional layers are responsible for feature extraction, allowing us to capture certain patterns that help us identify key features indicative of pneumonia, such as opacities or abnormal textures in the lung region. Max-pooling layers were used to help in reducing the spatial dimensions of our dataset, and as a result, lowered the overall computational cost and prevented overfitting. Dropout layers on the other hand, helped prevent overfitting by randomly dropping neurons during training.
On the other hand, our VGG16, ResNet, and DenseNet models leveraged pre-trained weights, a new classifier, and fine tuning of the final layers so that it could adapt to our specific task of identifying pneumonia. We believe this approach benefits from the large and extensive training on the ImageNet dataset. Our models were implemented using TensorFlow and KerasAPI, and Gradio was used to create an interactive web interface that allowed us to test our models sufficiently. In addition to this, preprocessing steps, such as normalization and resizing, ensured uniformity and compatibility with the model's input requirements.

CNN from scratch

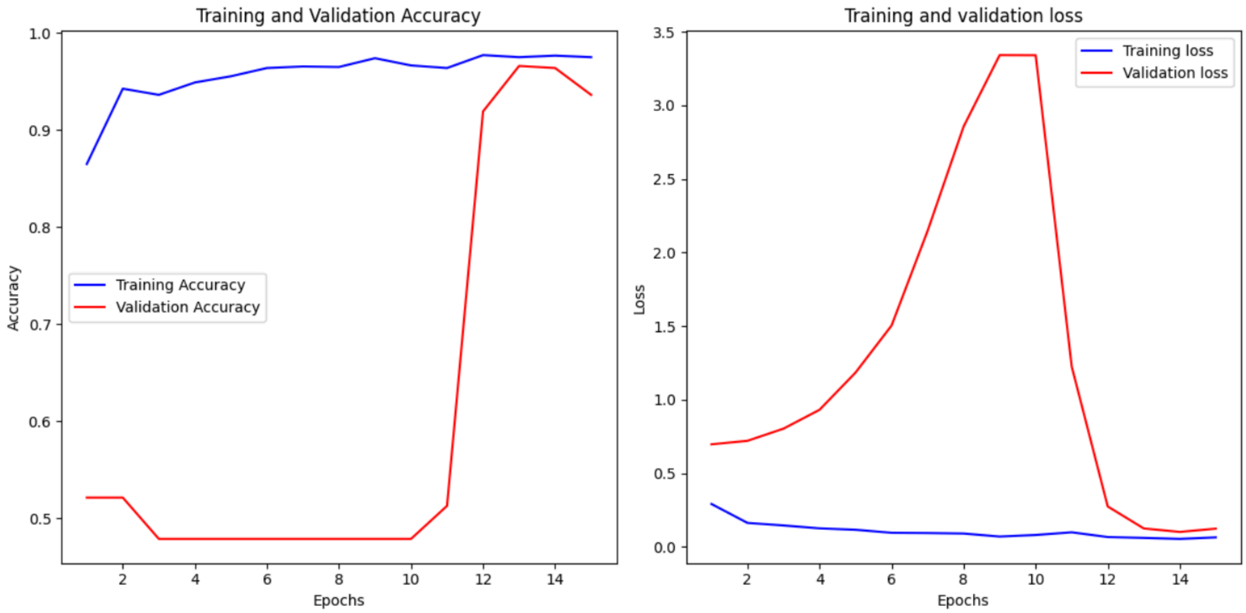

Figure 1.

The training accuracy of our model shows a steady increase from the very start, reaching nearly 1.0 (100%) by around the 8th epoch, indicating that the model is learning the patterns of our dataset effectively. However, the validation accuracy on the other hand starts at a low 0.5 (50%), and remains this way until about epoch 11 where it suddenly experiences a significant rise to about 95%. Unfortunately, it seems that the accuracy drops at around the 13th epoch. Suggesting potential overfitting where the model performs well on training data but poorly on validation data.

The training loss plot mirrors the expected results of a sufficiently trained model; we see that training loss decreases at a steady rate, which is expected as our model learns and adjusts its weight. By the 10th epoch, our training loss approaches 0, indicating to us that the model is fitting the training data effectively. In contrast, the validation loss shows an initial increase peaking at about the 9th epoch, but then proceeds to sharply decrease.

<u>**VGG16**</u>

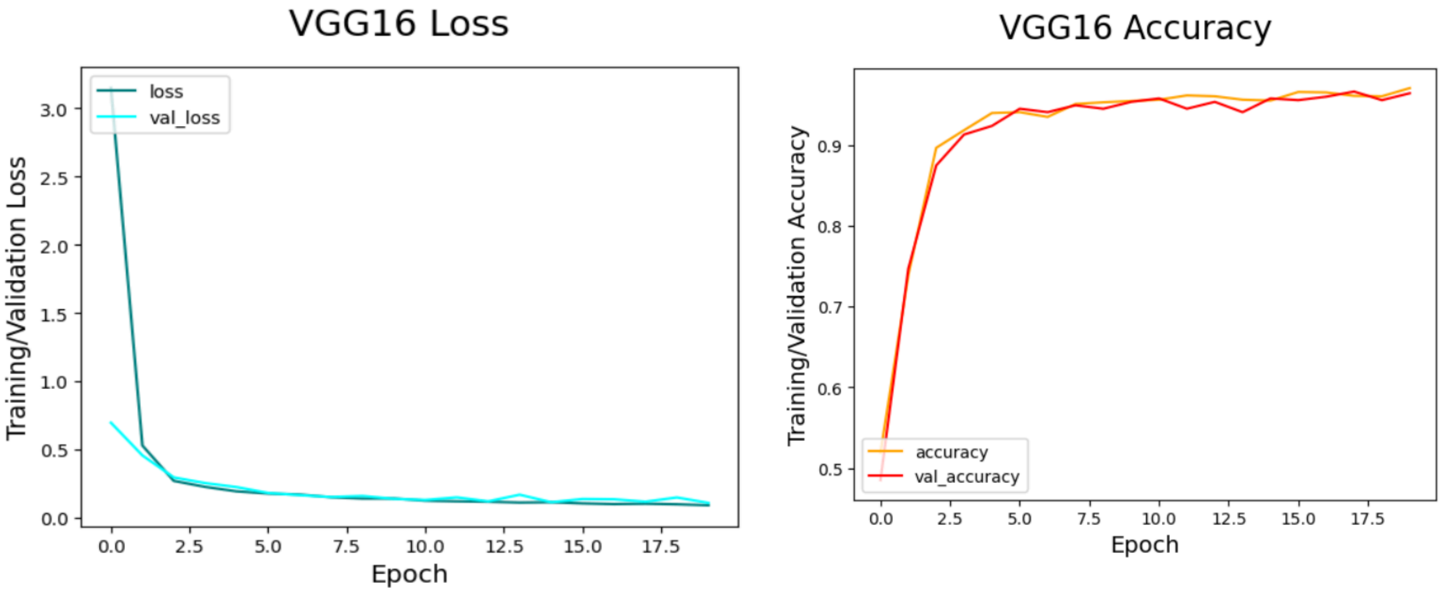

Figure 2.

Figure 2. illustrates the performance of the VGG16 model and shows a rapid decline within the first few epochs. Both the training loss and validation loss start at a relatively high place, with the validation loss being lower at first. However, by the 2nd epoch, both losses drop significantly and approach near zero for the remainder of the training process. This rapid convergence suggests that the VGG16 model, benefiting from its pre-trained weights, quickly adapted to the new dataset, minimizing both training and validation loss. The accuracy plot reveals that both training and validation accuracy start at around 50% but rapidly increase to nearly 100% by the second epoch, indicating that the model is able to effectively generalize well to the validation data, without significant overfitting, in contrast with our CNN model.

ResNet50V2 & DenseNet121

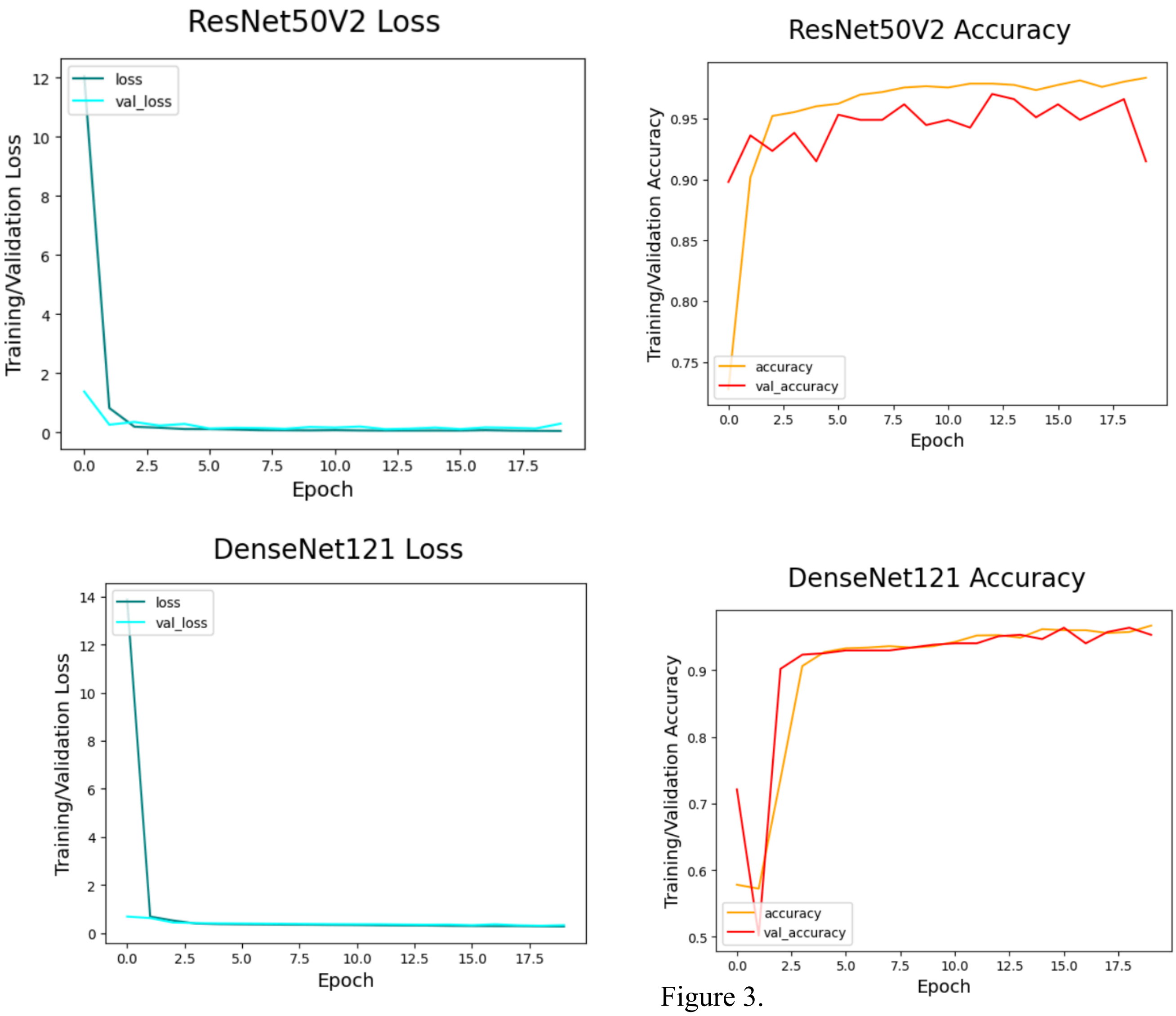

Figure 3.

Similarly, both the ResNet and DenseNet models demonstrate strong results when compared with its VGG16 counterpart. However, we can see the validation accuracy for the ResNet model in Figure 3 fluctuates and never stabilizes, although still maintaining high levels before dropping by around epoch 17. This suggests that while the model learns effectively, it may have trouble with generalizing unseen data, and can hint towards slight overfitting. However it is important to know that ResNet, and its architecture, can sometimes be sensitive to the specifics within a dataset and can be highly influenced by subtle differences in the validation images (*ResNet, 2024*)

In conclusion, the use of artificial intelligence in medical diagnostics, specifically for the detection of pneumonia as was done in our case, presents an opportunity that addresses the current limitations of contemporary diagnostic methods. This project has shown that the use of machine learning models, such as CNN's, VGG16, ResNet, and DenseNet can achieve impressive results with high accuracy, ultimately improving patient outcomes. Our models have shown the capability to generalize well on data previously not encountered, and although it is not accurate enough to be used as a stand alone procedure, they offer a viable solution that should be used with other methods of diagnostics, allowing for a reduced risk of misdiagnosis and delayed treatment.

How to download, configure, and train the 3 Pre-Trained Models

1. To use VGG16, ResNet50V2, and DenseNet121 in your program, you must import them using the Tensorflow.Keras library for it's layers, optimizers, regularizers,

ImageGenerator function, and other helpful functions. We also import Tensorflow and Numpy for data augmentation.

2. Next, you need download the database, if you have it on your device, then simply import it into your environment, otherwise refer to database downloading instructions on your database's website.
3. Now, that you have your database downloaded into your environment, you need to make sure that the images meet the requirements of the models. For us, all 3 of the models required the images to be (224, 224) in size. So we had to resize all images using data augmentation methods. You can also add other augmentations to your training dataset make the model training more efficient but not to your validation dataset.
4. Now, you must initialize the models and load the weights. If you have one of your own, use it, else you can use the weights from ImageNet which is a large visual database consisting of 14 million images and a thousand object classes. After initializing the model, you need to freeze it's layers and add your own on top. Refer to our **"3-pre-trained-CNN-LungDisease.ipynb"** file to see how we froze our layers and added custom layers on top.
5. Once you're done adding your custom layers, compile the model and train it using the your training set and validation set. Set your epochs to 15-20 because pre-trained models learn and start recognizing patterns very quick especially with ImageNet weights.
6. To test your newly trained model, you can create a Gradio Interface. You should customize it to your use and add your own functionality to it like we did. Refer to **"CNNmodelsPredictInterface.ipynb"** to see how easily you can create a working Gradio interface for yourself.

Github Repo: https://github.com/itsSuryanshu/CNN-models-LungDisease-ProjectCP468

Video Link:

https://drive.google.com/file/d/184yowdqyqvhIg64rnasxwtvu2pBmcHDE/view?usp=sharing

Citations

Cilloniz, C., Martin-Loeches, I., Garcia-Vidal, C., San Jose, A., & Torres, A. (2016). Microbial Etiology of Pneumonia: Epidemiology, Diagnosis and Resistance Patterns. *International Journal of Molecular Sciences*, *17*(12), 2120. https://doi.org/10.3390/ijms17122120

Gomes, R. F. T., Schuch, L. F., Martins, M. D., Honório, E. F., de Figueiredo, R. M., Schmith, J., Machado, G. N., & Carrard, V. C. (2023). Use of Deep Neural Networks in the Detection and Automated Classification of Lesions Using Clinical Images in Ophthalmology, Dermatology, and Oral Medicine—A Systematic Review. *Journal of Digital Imaging*, *36*(3), 1060–1070. https://doi.org/10.1007/s10278-023-00775-3

Heitz, M., Albrice Levrat, Lazarevic, V., Olivier Barraud, Bland, S., Emmanuelle Santiago-Allexant, Louis, K., Schrenzel, J., & Hauser, S. (2023). Metagenomics for the microbiological diagnosis of hospital-acquired pneumonia and ventilator-associated pneumonia (HAP/VAP) in intensive care unit (ICU): a proof-of-concept study. *Respiratory Research*, *24*(1). https://doi.org/10.1186/s12931-023-02597-x

Mehrparvar, Fatemeh. "Lung Disease." *Kaggle*, 9 Mar. 2024, www.kaggle.com/datasets/fatemehmehrparvar/lung-disease/data.

Nelson, Joseph. "How to Train a VGG-16 Image Classification Model on Your Own Dataset." *Roboflow Blog*, Roboflow Blog, 9 Apr. 2024, blog.roboflow.com/how-to-train-a-vgg-16-image-classification-model-on-your-own-dataset/.

Pates, K. M., Periselneris, J. N., & Brown, J. S. (2023). Pneumonia. *Medicine (Abingdon. 1995, UK Ed.)*, *51*(11), 763–767. https://doi.org/10.1016/j.mpmed.2023.08.003

Pykes, Kurtis. “Fighting Overfitting with L1 or L2 Regularization: Which One Is Better?” *Neptune.Ai*, 4 Aug. 2023, neptune.ai/blog/fighting-overfitting-with-l1-or-l2-regularization.

Simonyan, K., & Zisserman, A. (2015). Very Deep Convolutional Networks for Large-Scale Image Recognition. International Conference on Learning Representations (ICLR)

Team, Keras. “Keras Documentation: Save, Serialize, and Export Models.” *Keras*, keras.io/guides/serialization_and_saving/. Accessed 2 Aug. 2024.

Team, Keras. “Keras Documentation: DenseNet.” *Keras*, keras.io/api/applications/densenet/#densenet121-function. Accessed 31 July 2024.

Team, Keras. “Keras Documentation: Resnet and RESNETV2.” *Keras*, keras.io/api/applications/resnet/. Accessed 31 July 2024.

Team, Keras. “Keras Documentation: The Sequential Model.” *Keras*, keras.io/guides/sequential_model/. Accessed 28 July 2024.

Team, Keras. “Keras Documentation: Image Classification From Scratch.” *Keras*, keras.io/examples/vision/image_classification_from_scratch/. Accessed 28 July 2024.

“What Are the Advantages and Disadvantages of Using Resnet-50? | 5 Answers from Research Papers.” *Scispace*, typeset.io/questions/what-are-the-advantages-and-disadvantages-of-using-resnet-50-1cx7wvczbf. Accessed 1 Aug. 2024.

“Activation Functions in Neural Networks [12 Types & Use Cases].” *V7*, www.v7labs.com/blog/neural-networks-activation-functions#:~:text=The%20linear%20activation%20function%2C%20also,Linear%20Activation%20Function. Accessed 2 Aug. 2024.

ResNet: Residual network - javatpoint. www.javatpoint.com. (n.d.). https://www.javatpoint.com/resnet-residual-network#:~:text=Its%20residual%20connections%20enable%20better,to%20overfitting%2C%20and%20limited%20interpretability.