# CSC 434: Artificial Intelligence - Pneumonia Detection

# Introduction

The project that we have selected to extend is one that leverages AI to detect pneumonia in X-ray images. Pneumonia is most diagnosed using chest X-rays. Pneumonia is a condition in which one or both lungs fill with fluid. X-rays work due to this fluid inducing a reflection of X-rays that result in “white” or solid (e.g. bones) regions in the X-ray film. A doctor diagnoses pneumonia by examining these X-rays visually, comparing the percentage of healthy lung (which X-rays pass through and do not reflect) vs unhealthy lungs where there is a larger area of reflection or “white”. These comparisons can take time, are prone to error, and in some cases may benefit from a second opinion. We are presenting this diagnostic tool, which leverages machine learning, to facilitate this. This is a practical application of machine learning, and through the addition of a GUI interface that is intuitive and easy to use, it is our hope that our contribution will be seen as a positive one.

# Methods

We are extending the project outlined here: [Pneumonia Detection using CNN with Implementation in Python](https://www.analyticsvidhya.com/blog/2020/09/pneumonia-detection-using-cnn-with-implementation-in-python). This project utilizes a Convolutional Neural Network, or CNN, to analyze the X-ray images. We have a dataset and code already, so our plan is to improve upon what's already there. We will experiment with different settings with the CNN and evaluate the model using different sizes for the photos. Additionally, we intend to utilize different algorithms for training our model, specifically relu and sigmoid. Lastly, we will look at all other functions and see if they have different settings that we can use. If we find that they do have settings we can change, we will test out the results of those settings with the best version of our prediction model. Through the addition of a GUI interface, where a user, such as a doctor, would be able to select an X-ray image and the GUI will run the image through the neural network. The program could then display its prediction to the user. This task is not a simple one, so of course we need to use libraries to help us solve the problem. The libraries that we plan to use for the neural network are keras, and sklearn. To help us with saving images we will be using the pickle package. To help us with dealing and plotting the data we will be using numpy, matplotlib, and seaborn. For our GUI we are using tkinter or Qt Designer and PIL to display image files. We will also use cv2 to load the images. Lastly, we use tqdm for a progress bar so we can see what epoch we are on for the neural network training. The project will present challenges along the way, but we believe we have selected the proper methods to complete the project.

# Dataset

The dataset selected for our base project has already categorized all the images. Additionally, the dataset was pre-split into train, test, and validation sets. The dataset also has three different classifications which are Normal, Viral pneumonia, and Bacterial pneumonia. To increase entropy, we are performing image transformations to discover if this increases our model’s accuracy.

# Goal

Our goal is to improve the accuracy of the AI and implement a GUI to serve as a usable tool. We have decided to keep our goal to improving upon the AI as much as possible and add a GUI interface to interact with. Most implementations of AI go through the training aspect of the neural network. We wanted to go a step farther and figure out how to save and load different models into an application. We figured this would be a very real-world situation in which we train the AI as much as possible and then release the GUI for someone to use. Once someone uses the program, it’s bound to need updates. So, we could over time get more training data to train the model. If that were to happen, then we would know exactly what to do to update the program. Overall, our goal is to try and improve the already existing AI’s accuracy and implement a way to check single images with a GUI interface.

# Plan

After establishing our goals, we have devised a plan to achieve our desired end results in an orderly and timely fashion. Our first step is to get the base program runnable. Once we do that, then we can record the results. After getting the results, we will need to analyze them, and that will be our benchmark to improve upon. Once we do that, our first plan is to run a different algorithm for the training process. This will give us another record of results that we can then compare with and see which is better. Our next idea is to change the size of the images, as well as changing the variables for the CNN. After comparing that, we can see what other functions are being used and see if there are any other options that we can run them with. Following this, we should have a pretty good idea on which settings improved the AI the most. We can then do our last test which is putting all our best settings to use by essentially testing all of them together. We feel that this will be enough of a workload to take up the allotted project time, but if there is more room for research and practice, we can try a different package to create our CNN and repeat the same process.