Bomher Sebastian

**Comparing Machine Learning Methods**

-Supervised-

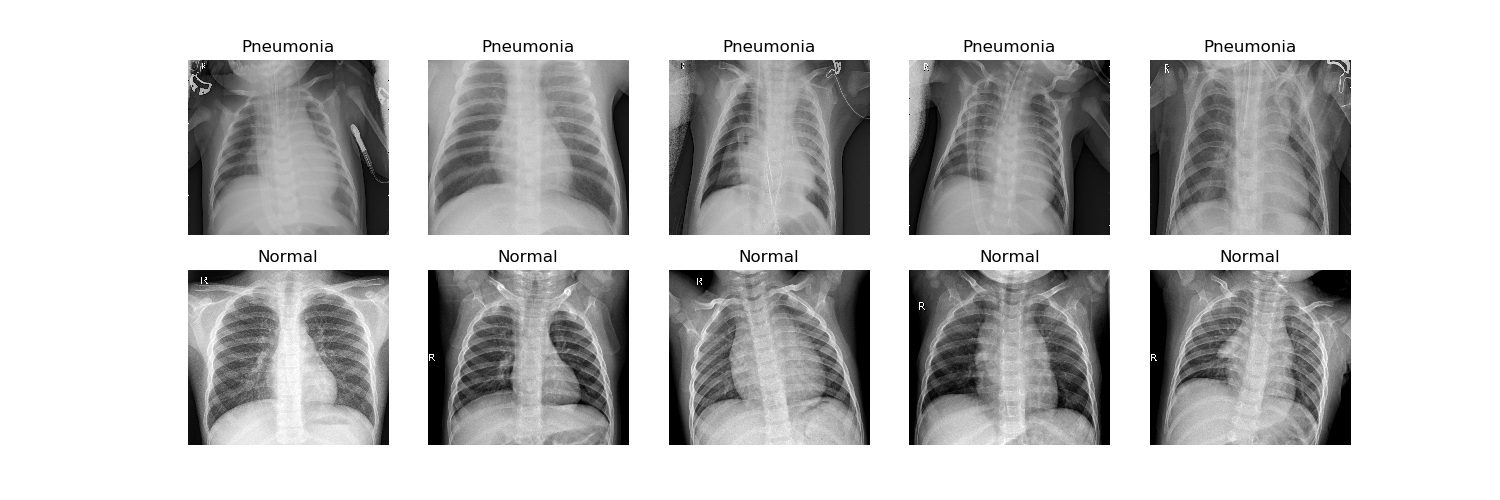
Machine learning is the scientific study of algorithms and statistical models that computer systems use to perform a specific task without using explicit instructions. There are 2 types of machine learning methods: supervised and unsupervised. In this paper we will be comparing two supervised methods on one dataset to compare their performance, speed, accuracy and any additional information to verify which is better.

The methods we will be comparing are Convolutional Neural Networks (CNN for short) and k Nearest Neighbors (kNN for short). For now on in this paper we will use the short version.

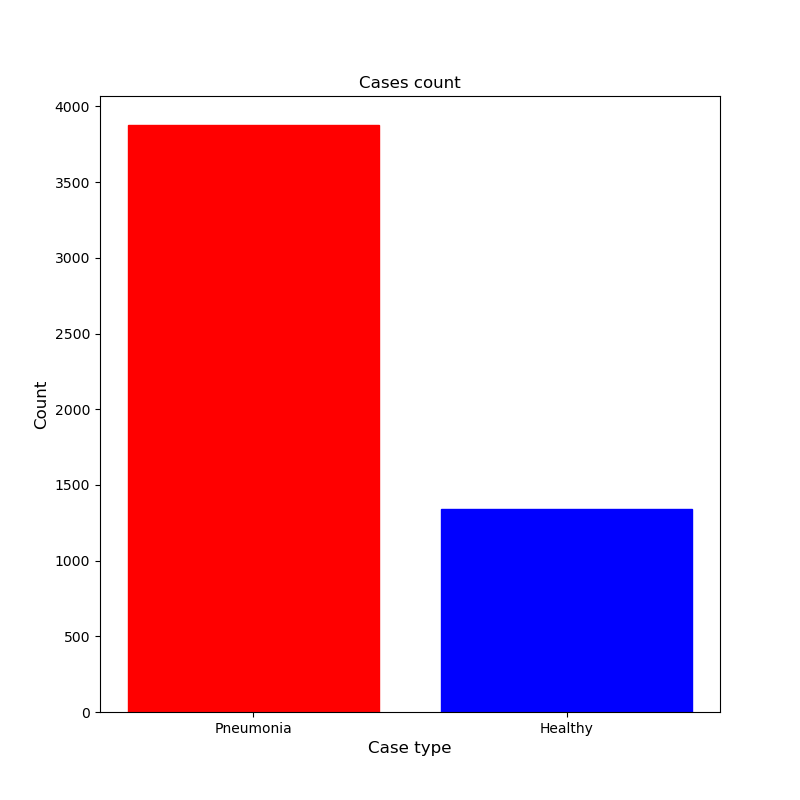
1. **Dataset**

Our dataset is called Chest X-Ray Images (Pneumonia) created by Paul Mooney and can be found on kaggle on the following link: <https://www.kaggle.com/paultimothymooney/chest-xray-pneumonia>. It has a total number of 5863 images contained in 2 categories: healthy(labeled as Normal) and Sick (labeled as Pneumonia). Our dataset is split into 3 folders: train, test and validation, each containing 5216, 624 and 16 photos. Each folder is split in its 2 categories mentioned earlier. This means we don’t have to split our dataset for testing and validation. Each picture has a size of approximately 250 kilobytes with a width of average 2000 and a height of average 2000.Our dataset has roughly 1 GB. The size of each photo is pretty big, later on we will shrink their size to keep only the most important features as it is a computational problem to work with such a size.

Lets take a look at some sample photos.



Our healthy and pneumonia case looks like this:



As we can see it is highly unbalanced, as it is with most medical cases. We can observe that we have almost three times more pneumonia cases than healthy cases.

Our goal here is to build a model that can predict whether or not a picture with a chest x-ray has pneumonia or not.

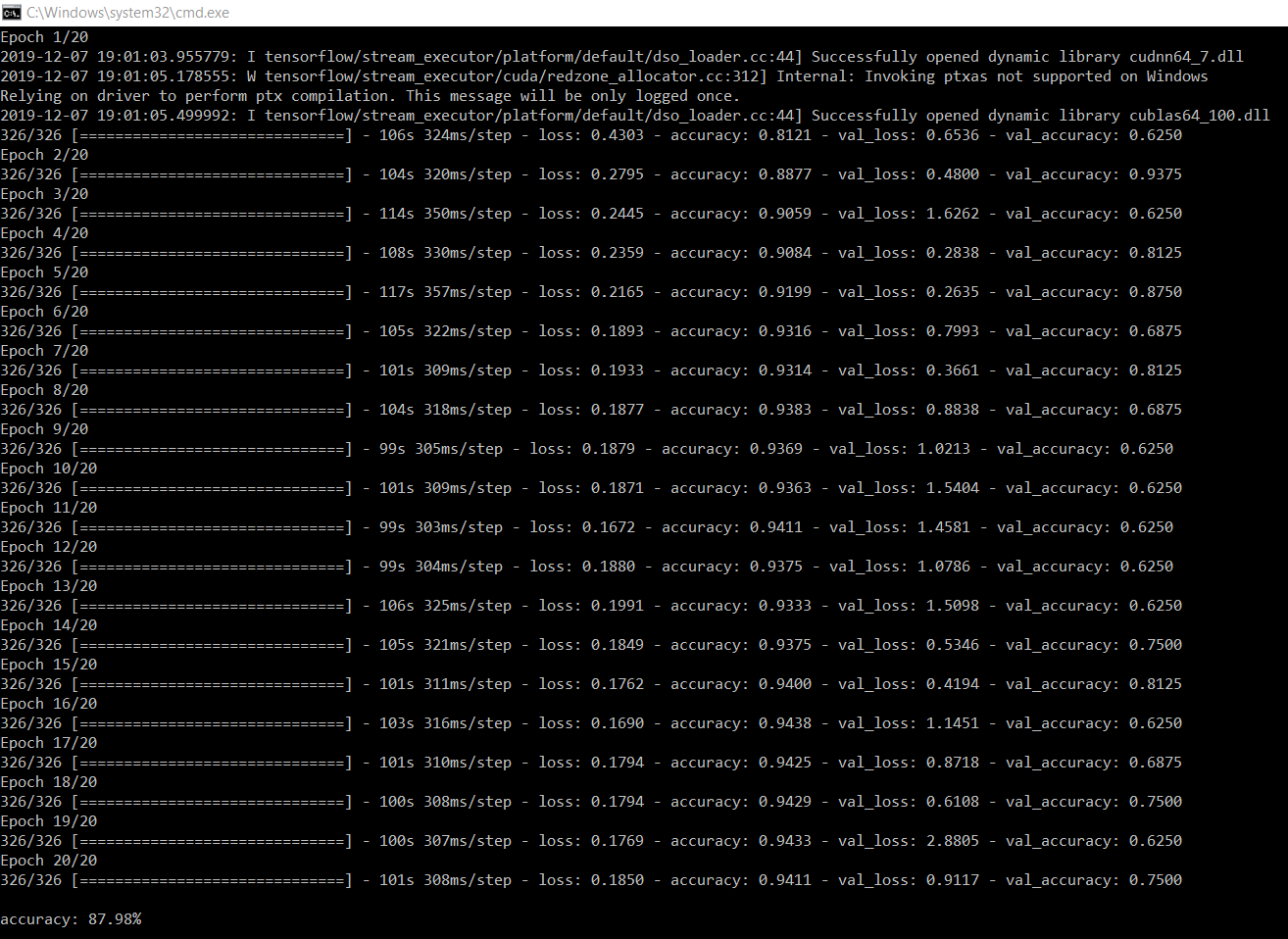
1. **Convolutional Neural Networks**

Our first method to inspect is CNN. To understand this method we will talk about Neurons in Machine Learning. Neurons are inspired by human brain. A brain neuron function is to process and to pass on information to other neurons, so a neuron in machine learning receives an input, processes its information, and passes it as an output to another neuron. Each neuron has a weight and a bias. A neural network is composed of multiple neurons interconnected to another set of neurons which in their turn pass their information to an output to observe some data. A deep learning neural network means that there is at least one or more layers between the input layer and the output layer. CNN takes deep learning and makes the neurons weights into a matrix. Most commonly used shapes are identity shapes, either 3x3 or 5x5.

To implement our CNN we will use keras and keras-gpu, an deep learning framework made for python. One of the problems with CNN is that it requires huge computational power, as the total number of trainable parameters can sometimes reach to millions. To fix this problem python allows to use GPU, as it uses multi-threading computing, unlike CPU which is linear. GPU computing can be 8 times as fast as a CPU. The only GPU supported by python is NVIDIA, the alternative being colab-research, a ipython notebook based website with GPU support.

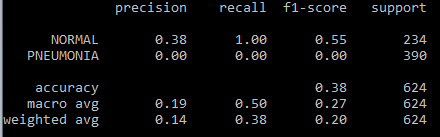
First thing to consider is the image size. Due to its high size it will be impossible to train our model. For this reason each image is shrunk to a size of 255x255

Our model will be composed of 3 modules, each containing 32 layers of convolutions, followed by an activation function of type rectified linear unit (ReLU for short) followed by pooling. At the end it those modules we flatten the result, add a fully connected layer, activate with ReLU, dropout half to prevent overfitting add another fully connected layer, finally activate with a sigmoid function. The total number trainable parameters is about 1.2 million. This model takes a batch of 16 photos to learn at one time, and it does it 326 times to pass through all training photos. At the end it will check with another set of photos called the validation set to evaluate the current model. This step is called an epoch, at it will be repetead 20 times. Each epoch takes approximately 100 seconds to complete with around 300 ms per step. This takes a total time of about 30 minutes to train.



In the picture above we can observe our trained model and its process.

Below we can observe the confusion matrix:



Our model has an accuracy of 87.98% on the testing set. This is a pretty good result, considering that this model is not so big compared to VGG16 or ImageNet, which has hundred of millions of parameters. Although they take more time to train their results are better, some of them reaching 99%. As the accuracy increases the time to train a model increases exponentially. It is a trade-off to balance depending on your requirements.

1. **k Nearest Neighbors**

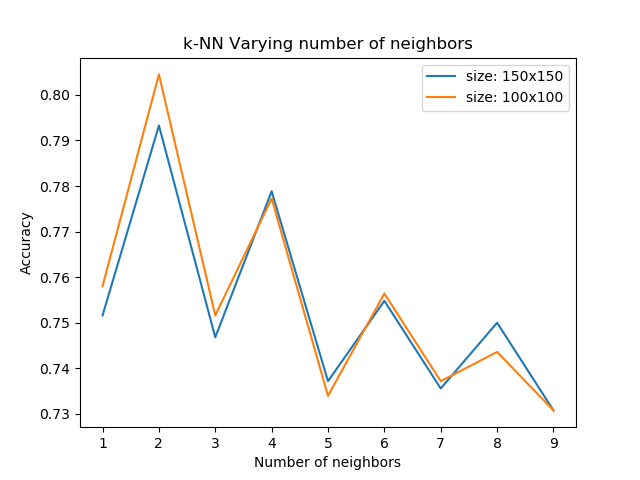
Our second method to perform is kNN. kNN takes the input and places it in an 2-dimensional space based by its features. After its training it will take its testing set and determine the k nearest point to determine which class it belongs to. Our distance can be calculated in 3 ways: The Euclidian distance, the Manhattan distance or the Minkowski distance.

Given a number k it will calculate the closest k neighbors, take their classes, and the majority class will be labeled to the original point.

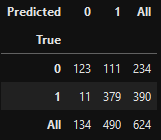


As out previous method, our images are too big to be processed, so they will be shrunk to 100x100 and 150x150 to test which one are the most suitable

First thing we will have to choose a suitable number for k. We will iterate through 1 to 9 to see which one is the most suitable one for our requirements. The results are the following:



We can observe that our most suitable k is 2 with a photo size of 100x100. We will train a 2-NN classifier the confusion matrix.



As we can observe it has a higher confusion for normal(labeled 0) to pneumonia(labeled 1) to a total count of 111 mismatched cases.

1. **Conclusion**

Both methods are their uses, kNN is faster and easier to understand but CNN is the better

method for image classification. Even though it requires high computational power it has a better accuracy and more flexibility in creating models for a specific classification requirement.