Aileen Dugan CSE 40535

Bone X-Ray Classification with MURA dataset using DenseNet169

The database I used for bone x-ray classification was the MURA-v1.1 database. This database contains training and validation images broken up by body parts including shoulder, finger, forearm, wrist, humorous, etc. The images are labeled by professional doctors as either positive for a break or abnormality, or negative for being healthy. The database as a whole has a total of 40,561 multi-view images from 14,863 studies of 12,173 patients. I used a subset of this dataset including only the shoulder images for both trainning and testing as the database was too large to load fully. The number of training images of just shoulder x-rays was 8,379 while the number of testing/validation images for only shoulder x-rays was 563. I believe that this subset of the data was not an ideal amount as 10,000 plus seems more proper for training a convolutional neural network however. I also think that while it should be fine to have such a low number of testing images as 563 since they should be classifiable for a well trained CNN, because my CNN was not able to be trained on a larger portion of images, some variations were not able to be picked up as well and applied to the testing images as well as I would hope. Thus, a trend of predicting a negative result on a testing image that was actually positive was seen. However, with more testing images, it is possible that this trend would be different and some more images could be incorrectly predicted as positive instead. However, the subset of training images would have been likely been sufficient for a different type of image other than bone x-rays. This is because of the complexity and also hue/coloring of x-rays as they are radiographic images. Thus, the way that radio waves pick up on the bones in images calls for some extra work.

The denseNet169 architecture as a whole, seemed to be the best way to handle such images as many MURA database users have used it because of its ability to handle vanishing gradients and handling of 3 channel images such as x-rays. It is also pre-trained on imagenet which helps to combat the lower number of images that desired in the x-ray database subset I used as well. DenseNet169 also has [6,12,32,32] layers. A picture of the general DenseNet architecture is shown in the following figure.

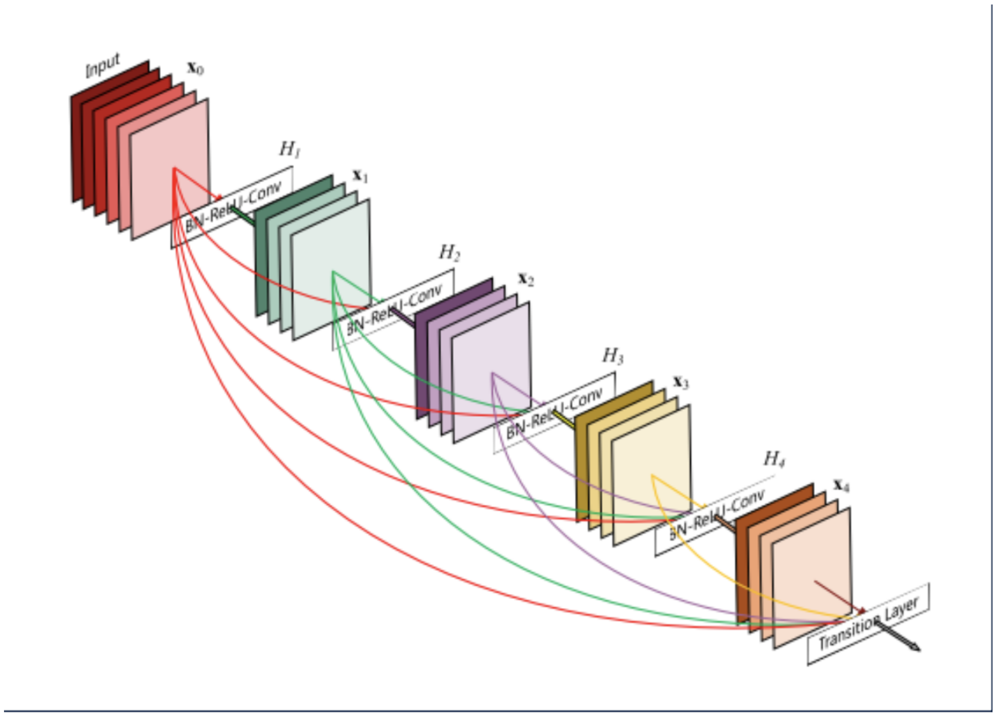

Figure 1: DenseNet architecture

Using DenseNet169 pre-trained on imagenet for my CNN model, the accuracy achieved on the test set was around 67% which was worse than the training accuracy of 83%. The accuracy going down in the testing is likely due to the subset of data being used to train the CNN not being large enough and therefor not containing the variation necessary to classify the testing images. This might especially be the case because the images are multi-view and thus not all images provide the same information about what a bone should look like or not look like and therefore there needs to be more images such that each view can be sufficiently learned for both positive and negative classification types.. The database also has some images of x-rays that are not exactly ideal as they do not all follow the same format. Some images are entirely of the x-ray while others are not. The differences of images including view, format, and hues can be seen in the following figures 2, 3, and 4.

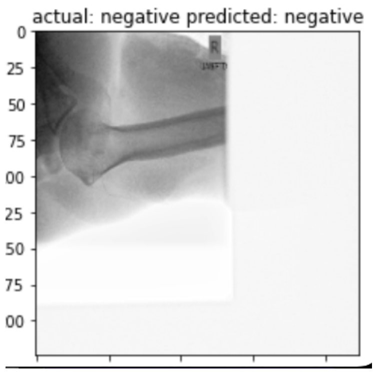

Figure 2: awkward formatting

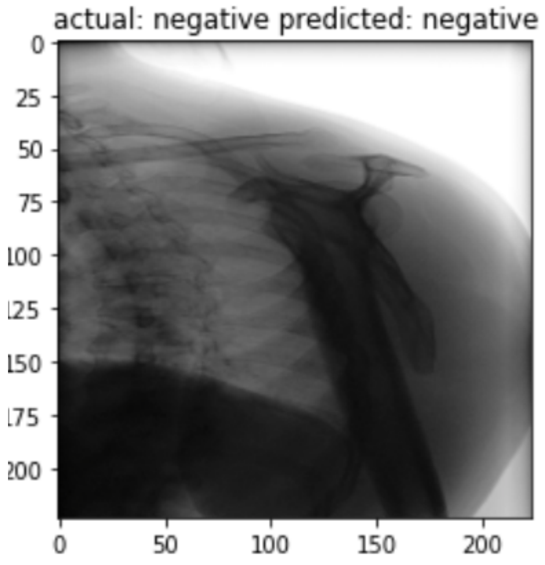

Figure 3: dark hues

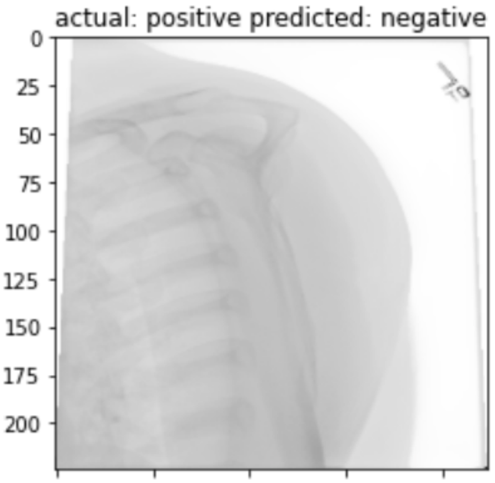

Figure 5: different view and lighter hues

As you can see, with this much variation in images besides just the bone differences themselves, it is necessary to have a larger set of training data to ensure that testing images can also be classified correctly. While DenseNet169 gave a training accuracy of about 83%, this means that it understood the training images that well but not necessarily testing images that might deviate or introduce a new angle, hue, format, or actual abnormality. I also think that since I could not grayscale all the images for DenseNet169, while this could have helped, it also could have caused issue as the x-ray images seem to need less variation in how saturated the coloring is for some rather than others. This might have needed some form of extra data preprocessing to normalize the coloring.