MSDS 458, Section 55, Assignment 4

Alan Kessler

Convolutional Neural Networks for Pneumonia Classification

The overall goal for this analysis is to build a convolutional neural network (CNN) that predicts whether a patient has pneumonia based on an X-ray of their upper torso. More important is the discussion regarding how such a model can be implemented and interpreted. Multiple model architectures are evaluated, resulting in a selected simple CNN from which credit assignment and features are obtained.

The data for the assignment is a set of almost 6,000 X-ray images. The images represent both healthy patients as well as patients with pneumonia Kermany, Zhang, & Goldbaum, 2018). The data is cited below and is not included with the assignment due to its large size and availability on Kaggle.com. Approximately 600 of the images are for testing. The validation directory only contains 16 images, so 20% of the roughly 5,000 training images are used for validation instead. The images intuitively have a single channel and vary in size as shown in Figure 1. The variance in size and orientation of the images present a challenge in creating a model that generalizes well. To allow for better generalization on a relatively small amount of training data, training images are randomly flipped, zoomed, or angled creating a more varied set. Keras allows the data to be loaded in batches and transformed in the same step based on directory structure. In this step the data is split into batches of 32 and resized to 128 by 128 pixels to balance predictive power with processing time considerations.

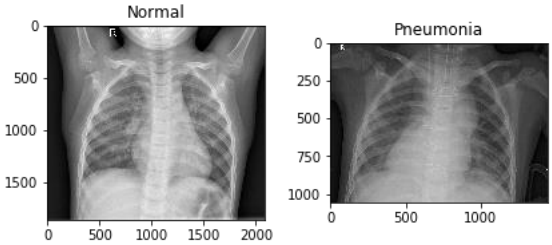

Figure 1 – Example Images

The first step after transforming the data is to specify each of the three model architectures. All three are based on the VGGNet architecture. That architecture built on the achievements of prior classifiers, but its defining characteristic is the use of consecutive 3x3 convolutional layers followed by a max pooling layer. The researchers behind that architecture found that the decrease in number of

parameters allowed them to implement a much deeper structure more efficiently (Simonyan & Zisserman, 2014). Drawing from their work, the three models represent three levels of simplification from the original VGGNet. The first model uses two 3x3x32 convolutional layers, max pooling layer, a 32-node dense layer, followed by a dropout layer with probability 0.5 before converging to a dense layer with sigmoid activation for classification. Model 2 adds a series of two 3x3x64 convolutional layers and a max pooling layer after the first max pooling layer. It also adds an additional dense layer with both dense layers having now 64 hidden nodes. Model 3 increases the complexity further by adding an additional series of two 3x3x128 convolutional layers and a max pooling layer after the second max pooling layer. It also increases the number of hidden nodes in the dense layers to 128. For all of the models, the default RMSProp optimizer is used to minimize binary cross-entropy.

The next step is to train the models. One of the features of Keras that I have found new to me is the ability to stop training early if the validation loss is not improved after a specific patience period. This can save time in training and avoid overfitting. The model that fits the validation data best is then the one saved. Despite the difference in number of layers, the number of trainable parameters and training durations are somewhat similar across models. That speaks to the efficiency that convolutional layers provide when identifying features in the data. Following training, the fit statistics are printed by epoch. The ROC curve is also generated for testing data along with a summary lift chart. The lift chart places test probability predictions in quantiles and shows the average response for each quantile. This chart is able to show how the model segments patients.

The results from the analysis above demonstrated that Model 1, with an area under the ROC curve of 0.904 compared similarly to the other models with AUCs of 0.922 and 0.930 respectively. Such a high AUC suggests that the model is ranking the patients well in terms of likelihood of having pneumonia. Model 1 is ultimately selected due to having a similar performance with a parsimonious model structure. The fit history by epoch for Model 1 are shown in Figure 2. The large spike in poor performance in epoch six for the validation data demonstrates the importance of the patience parameter when using early stopping. The ROC curve on test data is shown in Figure 3. Lift shows that for the top two quantiles in which the model is confident that the patient has pneumonia, the patient is in fact ill at a high rate.

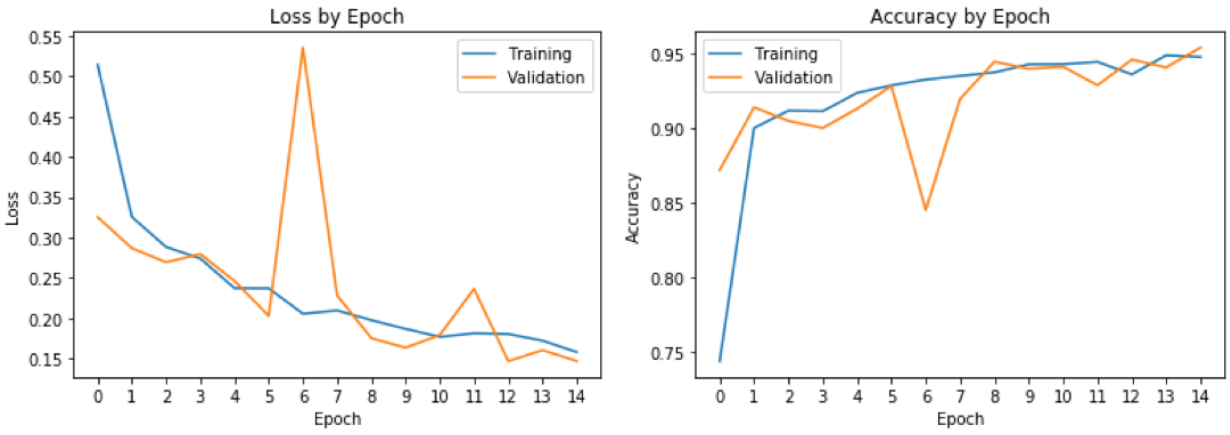

Figure 2 – Model 1 Fit History

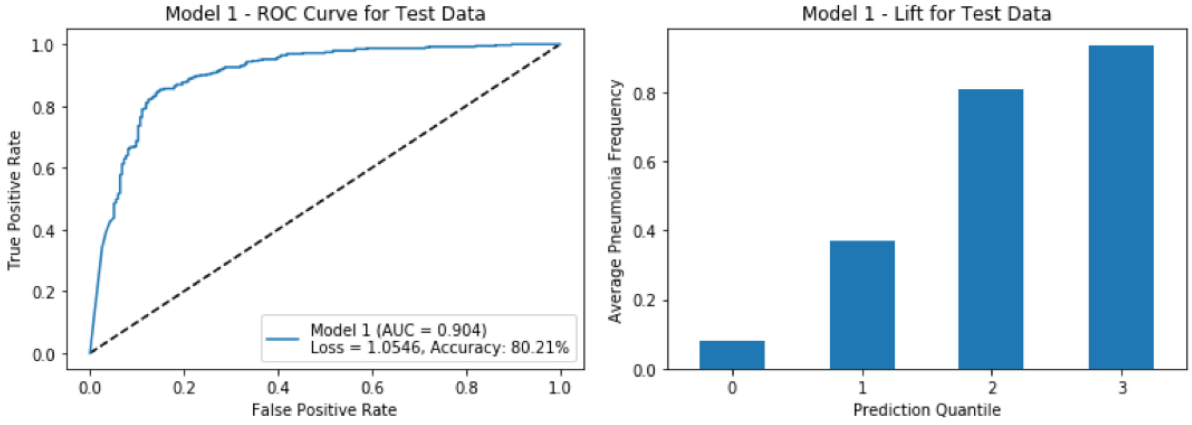

Figure 3 – Model 1 Test ROC Curve and Lift Chart (Upper Two Quantiles Combined)

The fit statistics and lift show that this model is very successful in classifying patients. The large values in the upper quantiles for lift suggest that in implementing the model, high values suggest a great likelihood for pneumonia. In these circumstances, a false positive is both unlikely and medically more conservative. These kinds of values would be useful as flags to medical professionals prior to an X-ray being more closely looked at. It could also serve a purpose in areas where medical resources are limited.

The model is less confident about values with a low likelihood. The ROC curve shows that a false negative rate, which could be a serious medical issue, could be a problem. By focusing more on the higher model scores, the model can serve to augment its human counterpart. Overall, the result could be a queue for X-rays where images that the model is confident about get a lower priority for a human to review but can also give medical professionals an early look into results. Finally, the model can serve as

a potential tool for what to look for when analyzing X-Rays. The convolutional layer features are visualized in Figure 4. While this help to interpret the layer's function, the Shap values discussed in the previous assignment are shown in Figure 5.

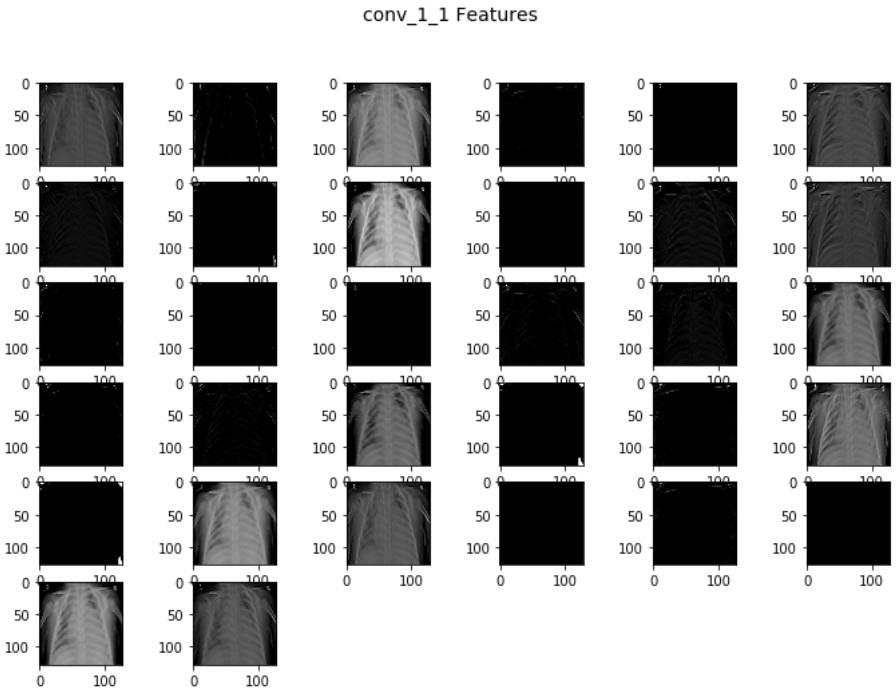

Figure 4 – Model 1 Convolutional Layer 1 Visualized

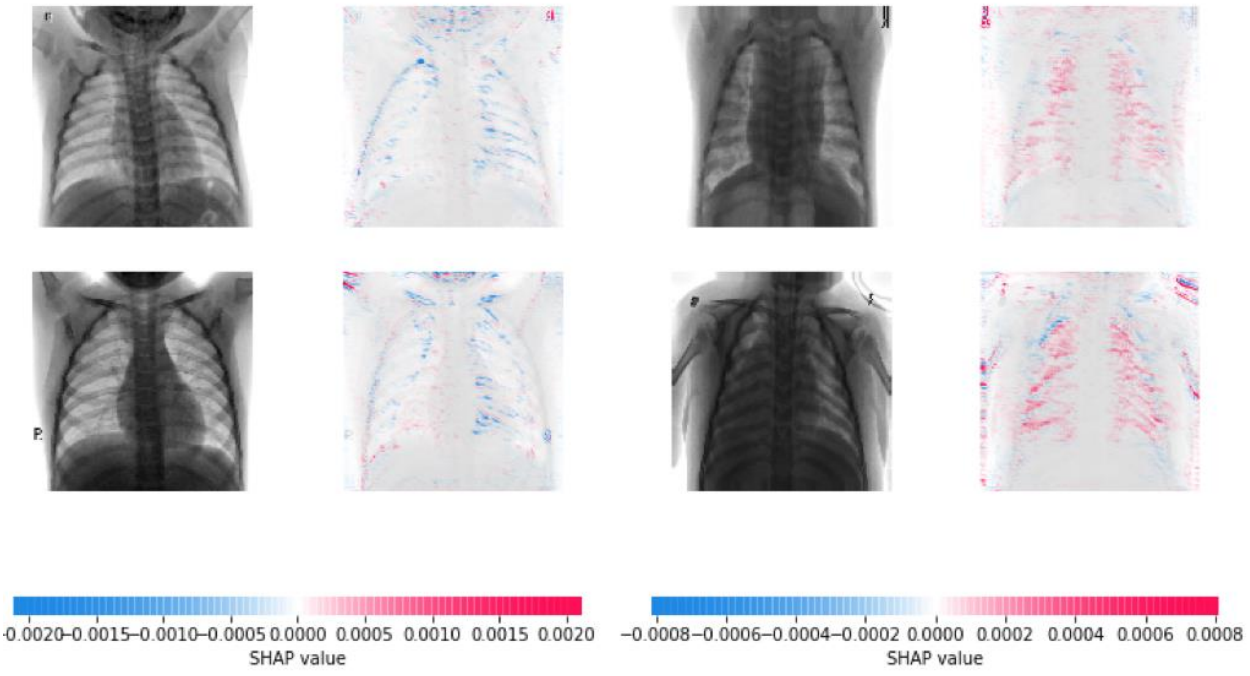

Figure 5 – Shap Values for True Negatives (left) and True Positives (right)

The Shap values show the parts of the image that contribute to one classification or the other. The examples above show that relatively high-contrast ribs of healthy patients result in a greater likelihood that the lungs are clear. The specific areas of interest are highlighted by the visuals.

In terms of programming, the models use the Keras python module with the TensorFlow backend. The ImageDataGenerator feature to feed images in via the directory structure and the ability to add early stopping call backs are valuable aspects of using a high-level API like Keras. Numpy, pandas, and matplotlib modules are used to generate and visualize fit statistics. The Shap module allows for robust credit assignment as explained above. The training itself was done on AWS EC2 instances that incorporate GPUs for improved computation times. The program, kessler_pneumonia.py, contains the scripts used in the analysis. One of the changes made for this assignment was implementing more of a narrative flow to the script that is reminiscent of the Keras documentation tutorials and fit the notebook style of development. The downside of this approach is that while it works from an exploratory standpoint, maintaining and implementing a more conventional Python script organization.

Based on this analysis, it is possible to use convolutional networks to automate aspects of a complex task. While these neural networks are complex as well, the ability to visualize a parsimonious model's layers maintains a level of interpretability. The next steps with this work could start with developing an implementation pipeline extending the use of AWS. From a modeling perspective, this analysis demonstrates the efficiency of convolutional layers. It would be possible as a next step to use higher resolution images and even deeper networks if that is something that the users of the model feel comfortable with.

References

Kermany, D, Zhang, K, & Goldbaum, M. (2018). Labeled Optical Coherence Tomography (OCT) and Chest X-Ray Images for Classification - Mendeley Data [Data is set] doi: http://dx.doi.org/10.17632/rscbjbr9sj.2 via https://www.kaggle.com/paultimothymooney/chest-xray-pneumonia

Simonyan, K., & Zisserman, A. (2014). Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*.