Multiclass Thoracic Disease Classification through Convolutional Neural Network

Kelly Chan
Electrical and Computer Engineering
Rice University
Houston, Texas
kc162@rice.edu

***Abstract*— Medical diagnosis through artificial intelligence has presented enormous potential, however many hospitals and clinical settings have yet to implement any deep learning methods. There are several reasons for this divide including lack of transparency, difficulty with reproducibility, among many others. This project will address these two specific issues using explainability techniques and a comprehensive overview of how to build and test this particular model, which is especially helpful for beginners or healthcare professionals who are interested in integrating machine learning into their clinical workflow.**

I. INTRODUCTION (*BACKGROUND & MOTIVATION*)

Physicians are overburdened with the increasing number of patients requiring care for thoracic diseases, which easily exceed two hundred distinct types. This leads to the potential for AI in clinical settings.

Image classification is one of the most fundamental tasks in the field of computer vision. It is the act of categorizing pictures using assigned labels from datasets of annotated pictures. Using pixel-level analysis the model, in this case a modified Convolutional Neural Network (CNN), makes predictions on which group the input belongs to. CNNs are a popular choice for handling large amounts of high-dimensional, spatial inputs, and are flexible across many datasets with varying types of data.

The NIH Chest X-Ray dataset will be used to explore the potential of disease recognition due to its quality and size. There are thirteen separate categories of thoracic diseases that will act as the labels for the model. More information about the dataset will be explored in the following sections.

Many prior studies using this dataset have only explored binary classification and have low interpretability for outsiders. They normally only select one disease, usually pneumonia or COVID-19, and rigorously train their algorithms on these to achieve high performance.

Additionally, most professional studies use larger, pre-trained models such as the VGG-16 simply due to the size of the dataset and its computational needs. [2]

The project will classify some of the most commonly identified illnesses using a simple CNN to a high degree of accuracy and produce visualizations to demonstrate how the model produced its results.

Machine learning exhibits profound potential in not only aiding physicians and radiologists, but also for improving patient care, allowing for more efficient and precise diagnoses.

A. *The NIH Chest X-Ray Dataset*

Dataset selection was the first and arguably one of the most important steps in this project. The NIH Chest X-Ray Dataset is a very popular information archive containing 112,120 X-ray scans from 30,805 unique patients. First introduced in 2017, it has become a staple in medical imaging training.

Each image is over 2000 x 2000 pixels and annotated by a medical professional. There are fourteen distinct disease labels along with a "No findings" header, which makes up 53.84% of the total dataset with a total of 60,361 cases. The image below shows the distribution of each original label, generated during the initial analysis.

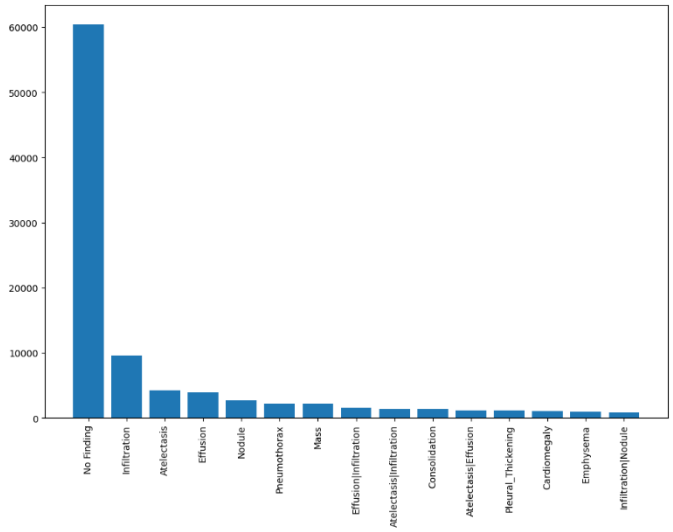

Fig. 1. Initial label distribution bar graph with 15 Labels.

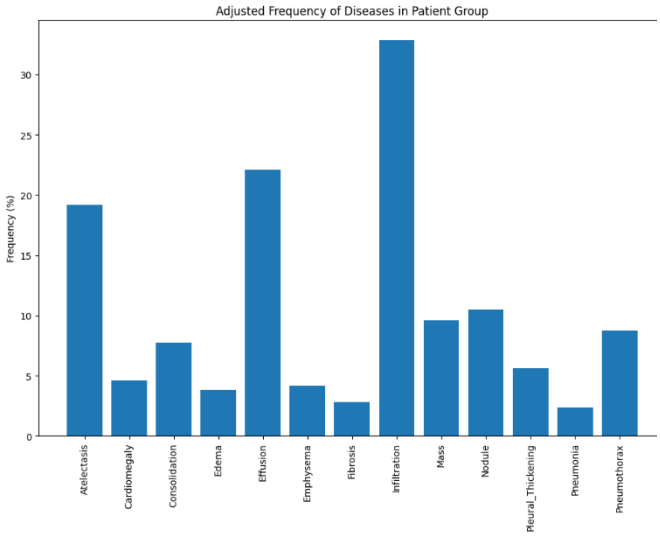

Fig. 2. Adjusted frequency of diseases after removing two irrelevant labels.

II. Experimentation

A. Model

While the dataset is well-annotated and abundant, this leads to a few issues. The computational power required to preprocess and train models is both immense and time-consuming. Therefore, a simplified neural network model was needed for this task. The Keras package from Python contained a pretrained MobileNetv1 that was customizable to specific needs.

MobileNet can be loosely defined as a reduced architecture CNN, which increases its efficiency and reduces its computational needs. Unfortunately however, there is a minor tradeoff for accuracy. The main principle behind its architecture was "to introduce a series of TensorFlow-based computer vision models that maximize accuracy while being mindful of the restricted resources for an on-device or embedded application." [3]

There are three main differences between MobileNet and standard CNNs that we'll examine below:

- The *first* is that MobileNet replaces standard convolutional operations with depthwise separable convolutions, as shown in Figure 3.

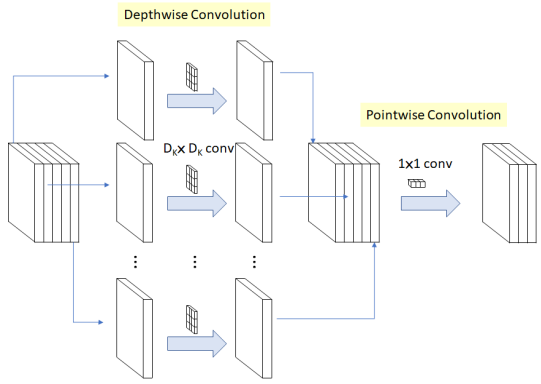

Fig. 3. MobileNet's depthwise and pointwise convolutional architecture. Adapted from [4]

- There are two main components of these convolutions: the depthwise convolution which performs spatial convolution independently for each channel, and pointwise convolutions which uses 1×1 convolutions to combine the output of the depthwise convolution across channels. These help shrink the number of parameters.
- The *second* difference is that MobileNet has a hyperparameter called the width multiplier, which allows the network to reduce the number of channels in each layer proportionally. This is beneficial for trimming down the model size.
- The *final* distinction is the resolution multiplier which reduces the input image resolution, decreasing the number of computations in the early layers of the network.

The MobileNet used in this project was built upon these three concepts but required further tuning to suit each experiment, of which there are four. The modified structure is displayed in Figure 4. In addition to incorporating more layers to the base model, this MobileNet utilized the Adam optimizer, binary cross-entropy loss, and an "EarlyStopping" function to prevent epochs from overfitting.

Layer (type)	Output Shape	Param #
mobilenet_1.00_128 (Functional)	(None, 4, 4, 1024)	3228288
conv_pw_13 (DepthwiseConv2D)	(None, 4, 4, 1024)	10240
conv_pw_13_relu (Activation)	(None, 4, 4, 1024)	0
global_average_pooling2d_3 (GlobalAveragePooling2D)	(None, 1024)	0
dropout_6 (Dropout)	(None, 1024)	0
dense_6 (Dense)	(None, 512)	524800
dropout_7 (Dropout)	(None, 512)	0
dense_7 (Dense)	(None, 13)	6669

Fig. 4. 40k iteration MobileNet layers.

B. Methods

The process from raw data to classification results comprises of five primary phases. The first and second stage were the most time-consuming due to the large number of datapoints.

- *Set-up*:
  - Download image zip file (42 GB) and CSV spreadsheet from NIH Chest X-ray website to local device. This process will likely span across multiple days.
  - Upload folders to Google Drive and mount them to Google Collab.

- Create a function to find all image paths and connect them from CSV to PNG pictures. Use the glob function to reach subfolders.
- Select the top 15 percent of individual labels which make up 84.27% of the total dataset. The remaining labels are combinations of independent labels, which are likely to confuse the model.
- Produce data visualizations (bar chart and frequency graph) for initial analysis. This is especially helpful for a broad overview of the dataset.

- *Preprocessing*:
  - Remove the "No findings" label. Any other label not under the 14 labels is classified as no finding.
  - Create one hot encoded columns for each unique disease label.
  - Drop diseases that have less than 1000 cases, of which there was only one label: hernia.
  - Enter weights for normalization.
  - Resize images: >2000 x > 2000 pixels → 128 x 128 pixels.
  - Split dataset into train data and test (75%) / validation data (25%).
  - Create data loaders for training. (Train and test data generators)
- *Feature Extraction*:

Use pretrained MobileNet model as a base for feature extraction.

- Load the model by calling MobileNet.
- Input the shape of the X-ray images and keep the rest of the default parameters.
- Add and remove layers of the base MobileNet model for feature extraction.

- *Model Training*:

There were four separate models with different samples sizes. The logistics of each will be explored in the following sentences.

- *1k:* The first test was unsuccessful. The model lacked a sufficient number of datapoints and was forced to end early due to random sampling. The data scarcity was caused by the number of labels (13). Because there were only 1000 images in this iteration, some of the diseases had very few cases to evaluate. Thus, this model was only able to complete three out of five epochs.
- *10k:* This trial was also suboptimal. Like the 1k sample, there were not enough images for the model to train and test on. Subsequently, it was only able to finish four out of five epochs.
- *40k:* This was the most successful model. Each sample size required different customizations. For this particular experiment, it was necessary to add one more depth-wise layer and use ReLu as the activation function.
- *60k:* A quick experiment using this sample size was attempted, however performance decreased. This may have been due to configuration issues.

While attempting a full trial with the full dataset, the file kept crashing, indicating the need for a more specialized machine. Additionally, the free T4 GPU provided in Google Collab was not enough to meet the processing requirements for any sample size above 1k. Purchasing the v2-8 TPU was essential to executing the program.

- *Explainability Techniques:*

Once the outputs are received, the next and final step is the application of explainability techniques. The two main frameworks used to improve interpretability are the confusion matrix and the occlusion sensitivity map.

Of the confusion matrices, there are multiple types used in this project: individual, macro-averaged, and aggregated.

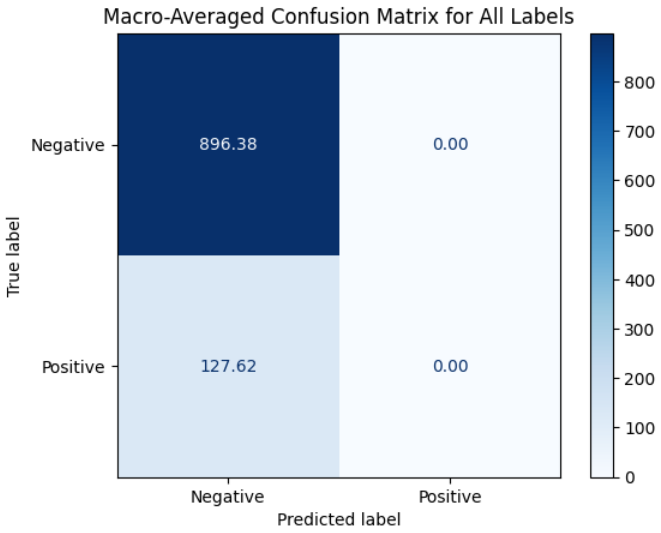

Fig. 5. 40k iteration Macro-Average Confusion Matrix.

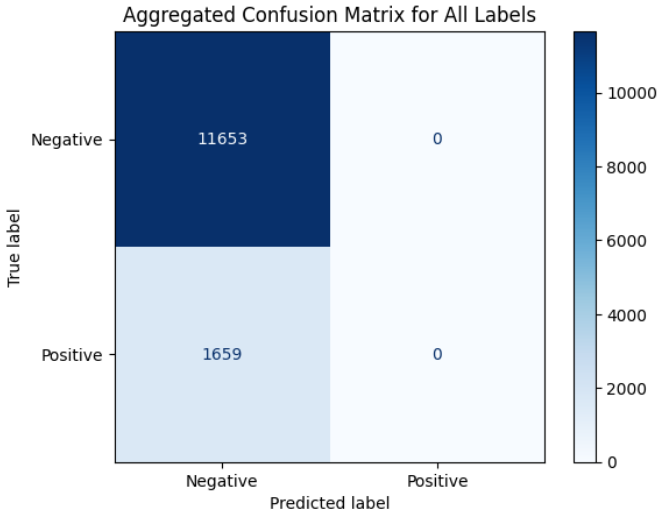

Fig. 6. 40k iteration Aggregated Confusion Matrix.

From the figures above, it is evident that MobileNet has low precision and recall for the positive classes, while it performs perfectly for the negative classes. This could be due to a severe data imbalance and model bias towards the majority class. Further tuning is required before deployment in real-world scenarios.

However, despite this, it is still able to achieve high scores for other performance metrics as demonstrated in the following results.

Next, the occlusion sensitivity map provides visual insight into MobileNet's decision-making. They are generally leveraged to conceptualize the most important areas the model uses for its predictions. This is especially useful for radiologists who have to manually inspect and annotate each X-ray.

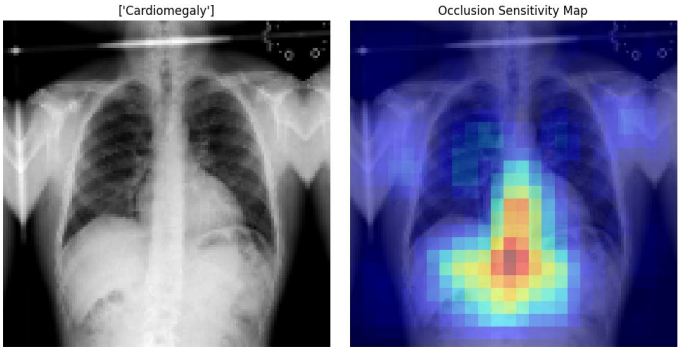

Fig. 7. 40k occlusion sensitivity sample output.

III. Results

The figures presented in this essay are all from the 40k sample experiment. However, each model was evaluated on its output metrics: loss, binary accuracy, mean absolute error, and AUC curve.

These were chosen based on prior research examples.

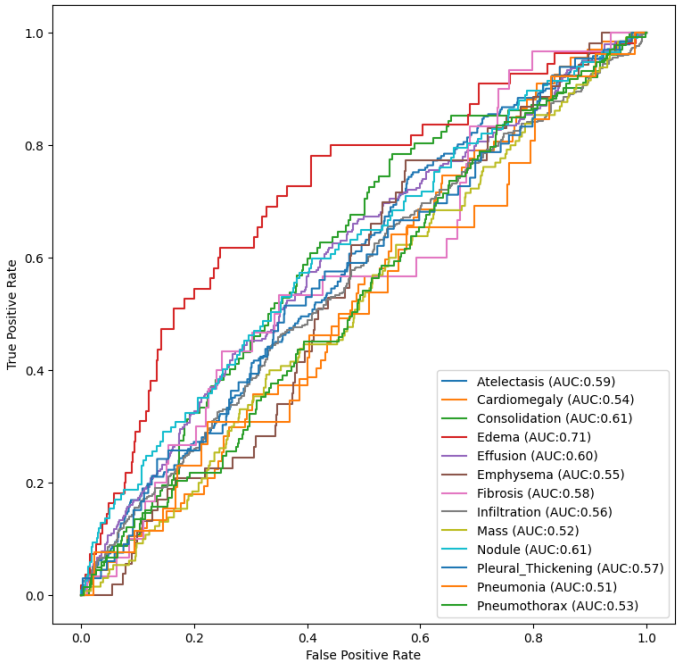

Fig. 8. Final AUC curve.

The 40k instance was able to achieve a binary accuracy of *.8767*, a mean absolute error *.2785*, and a validation loss of *13.30*, making it the best sample size tested.

The output also included a few sets of X-ray images with the doctor's diagnosis (Dx) vs. MobileNet's prediction (PDx). Many of them included multiple diseases, increasing the complexity of this endeavor.

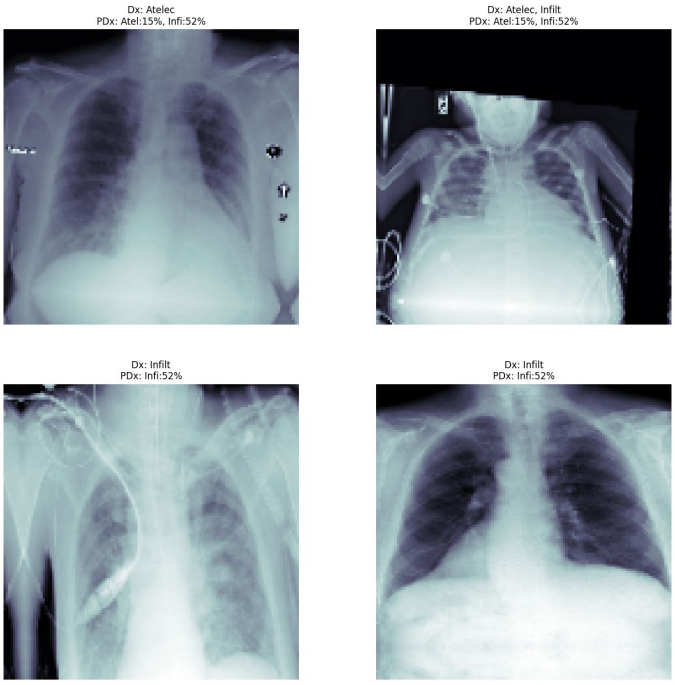

Fig. 9. Dx vs. PDx outputs.

More images can be found in the GitHub code link.

While the results are not yet acceptable for industry standards, this project presents a basic framework for more advancements. Occlusion sensitivity is a much less commonly utilized explainability technique and one of the only visual overlays that were compatible with MobileNet structure.

- *Failed Explainability Techniques:*
    - Many of the more prominent interpretability strategies were completely inoperable in this model. GradCAM, Shap Values, LIME, and Anchors were among the many that were attempted.
    - GradCAM is a prevalent standard for CNN models and was present in other studies using the same dataset. However, those papers employed the PyTorch module instead of Keras, indicating a possible incompatibility between either the package, the model or perhaps both with the implementation of GradCAM.
    - Despite these setbacks, these attempts provide worthwhile information on some of the explainability techniques available for MobileNet.

IV. Conclusion

The results were expected due to MobileNet's reduced architecture, and the code can easily be translated to accommodate more powerful models. More layers and samples correlated to higher performance but significant increases in processing times and memory usage.

The divide between the precision, recall, and accuracy scores was somewhat unexpected. This study highlights some of the limitations of a simplified model and the difficulty of handling imbalanced samples.

Experimenting with various model configurations and analysis tools helped optimize performance and human comprehension. While MobileNet minimized training durations, its inherent design was somewhat restrictive to what could be accomplished. Overall however, it is well-suited for the purpose of this project, which requires a lightweight algorithm that could handle significant amounts of data in a timely manner.

V. Future Work

If continued, the first logical step would be to optimize MobileNet to accommodate more sample sizes and discover more complementary explainability techniques. Furthermore, increasing parameter efficiency as well as attempting alternate sampling methods could contribute to higher performance metrics.

References

[1] *National Institutes of Health Chest X-Ray Dataset, "NIH chest X-rays," Kaggle, https://www.kaggle.com/datasets/nih-chest-xrays/data (accessed Dec. 17, 2024).*

[2] A. Ait Nasser and M. A. Akhloufi, "A review of recent advances in deep learning models for chest disease detection using radiography," Diagnostics (Basel, Switzerland), https://pmc.ncbi.nlm.nih.gov/articles/PMC9818166/ (accessed Dec. 18, 2024).

[3] "Google open sources MobileNetV3 with new ideas to improve mobile computer vision models," KDnuggets, https://www.kdnuggets.com/2019/12/google-open-sources-mobilenetv3-improve-mobile-computer-vision-models.html (accessed Dec. 18, 2024).

[4] S.-H. Tsang, "Review: MobileNetV1-depthwise separable convolution (light weight model)," Medium, https://towardsdatascience.com/review-mobilenetv1-depthwise-separable-convolution-light-weight-model-a382df364b69 (accessed Dec. 18, 2024).

[5] 2saad2, "-NIH-chest-x-ray-classification/train-simple-XRAY-cnn.ipynb at main · 2saad2/-NIH-chest-x-ray-classification," GitHub, https://github.com/2saad2/-NIH-Chest-X-ray-classification/blob/main/train-simple-xray-cnn.ipynb (accessed Dec. 17, 2024).

GITHUB LINK

https://github.com/Kelly-Chan2002/Multilabel-Thoracic-Disease-Classification-Through-a-Lightweight-Convolutional-Network