Diseases detection from Chest X-ray

Team 18: Danliang Wang[1], Youjung Kim[1], Brandon Leventhal[1]
[1] Master of Science in Computer Science, Georgia Institute of Technology

Abstract

X-ray is the oldest and most frequently used form of medical image, and still requires trained professionals to interpret the result, leading to high cost and human errors. We create a model that can automate the diagnosis of the 14 most prevalent diseases from chest X-rays. Our detection algorithm uses the largest publicly available chest X-ray dataset, called CheXpert, which uses Dense Convolutional Network with optimization strategies.

Our video presentation is available here.

Our project presentation is available here.

Our project coding is available here.

1 Introduction and Background

The problem we are studying is chest X-ray disease diagnosis. The problem is important because we want to automate the process of analyzing X-rays. X-ray is the oldest and most frequently used form of medical image, but it requires trained professionals to interpret the result, leading to high cost and human errors. Due to the low-cost and ubiquitous nature of X-rays it's easy to acquire data for experimentation and evaluation.

This problem lends well to a deep learning and big data approach due to the easy acquisition of data and the nature of the image processing. We believe that with deep learning, we can create a solution that reduces the cost and errors associated with human evaluation. We care about this problem because it can reduce medical cost and potentially prevent misdiagnosed diseases while allowing us to research and explore both deep learning and big data technologies.

The lack of datasets with strong radiologist-annotated ground truth and expert scores against which researchers can compare their models has been relieved by the release of the dataset from Irvin et al. [1]. Another important factor for this problem is improving the modeling techniques and was accomplished by Rajpurkar et al. [2] with a 121-layer Dense Convolutional Network (DenseNet) architecture providing detection rate at the level exceeding practicing radiologists. In order to reduce the noise often found in the images, Guan et al. [3] proposed a category-wide residual attention learning framework.

Apart from accurate classification of the pathology of a chest X-ray, it is also important to annotate the location of the pathologies in the image to provide rationale of the inference result. Li et al. [4] presented a network with CNN to predict both the labels and class-specific localizations. The localization awareness enables the model to produce better accuracies than state-of-art models. Ge et al. [5] then approached the chest x-ray classification problem by utilizing a new error function to mitigate the issue that others have had with multiple labels and imbalance in the data. Wang et al. [6] used a weakly-supervised classification to recognize and locate common disease patterns by using a combination of images and reports to produce their results as baseline for classifying the 14 diseases. The work done by these researchers has greatly advanced the classification of the 14 diseases, but there remains inconsistency which can be improved on.

2 Problem Formulation

This 14 diseases detection is a classification problem. We have created a model which uses a 121-layer convolutional neural network architecture that inputs preprocessed chest X-ray images and outputs the probability of 14 different diseases. Binary Cross Entropy with Logits loss function and Adam optimizer are accompanied to enhance the performance. Along with the results, we will produce a heatmap localizing the area of the image most indicative of the symptoms as well as ROC curves to show accuracy measures.

3 Experimental Setup

3-1 Data

For the data, we obtained the chest X ray data from CheXpert, a publicly available large dataset for chest radiograph interpretation released by J. Irvin et al. [1], 2019. It offers 224,316 chest radiographs of 65,240 patients labeled for the presence of 14 common chest radiographic observations. The observation result is provided in two csv files and marked the presence of 14 symptoms as present (1), confidentially absent (0), uncertainly present (-1), or not mentioned (blank) for each patient.

One interesting fact that we needed to consider was the uncertainty result indicated with -1. This decision was crucial as it decided the volume of valid training data. The uncertainty data was 16% of the trainable data (excluding the blank label) as well as 4% of total data. We decided not to ignore the uncertainty label because first it is a big loss on the volume of trainable data, and by losing the data it may cause a biased model. Although there are various methods, we used the very common and intuitive approach, data imputation. Specifically, one is to regard the "Uncertainty Present" as Present (U-Ones) and the other method is to take the data as Absent (U-Zeros).

3-2 Models

For the models, we utilized two models from Torch package, specifically DenseNet121 and ResNet101. Densely Connected Convolutional Network architecture is suggested by Gao Huang et al. [8] as a successful approach. We selected DenseNet 121 due to its improved retention of information flow and gradients throughout the network and eventually to reduce the impact of the vanishing gradient problem. We used the pre-trained densenet121 network to extract features and replace the classifier in the original model with a Linear layer and a Sigmoid activation function.

Residual Neural Network suggested by Kaiming He et al. [9] were used in this experiment. Non-pre-trained resnet101 architecture was employed to specify the output size of 14, the number of classes we have.

3-3 Software Stack

In this study, our experiments were conducted in a Python environment on an Ubuntu machine with GTX 1070 GPU in order to take advantage of the parallelization power by CUDA. We set up a virtual environment using Conda and installed all the required packages. For image pre-processing, from loading the images, resizing and normalization we used PySpark and Python Image Library to take the images as input. Torch and TorchVision were used not only to make use of the pre-trained models but also to train and test the models. To display experimental results, we have used sklearn to obtain the experiment measures and matplotlib to plot the data.

4 Approach and Implementation

The images were preprocessed such that they all have the same dimensions. We used Spark to create an ETL pipeline to preprocess the images and then feed the normalized images to Python to learn a Deep Learning model. We are using AUC and accuracy to evaluate our model.

4-1 Image Preprocessing

The first part of the project consisted of using Spark to transform our images into a standard format that could be fed into our classifier. We started this process by first doing a scan of the directory structure of the CheXpert study data set and getting an array of tuples consisting of a file path and file name. We then turned this array into a spark managed rdd so we could perform operations on the data in parallel to speed up the process because there are thousands of files in the list. We then performed a parallel map to walk through each patient's directory structure and retrieve the frontal image of the patient and create a new file path to save the processed image. This map produces a csv where each row provides info about what image needs to be processed and where to save this new image.

Once we have done our directory scan, we then move onto the processing. We take the csv produced from the previous step and use the built-in spark sql context functions to open it. This loads the csv file into a dataframe where we can easily perform parallel processing on it. The dataframe is then transformed into a rdd and we perform a parallel foreach to process each image. The foreach function takes the path of the image from the csv, opens it, then resizes the image to 320x320 and normalizes them with mean and standard deviation. The image is resized to fit the classifier that we are using in the next step. Once the resizing is finished, we then save the new image to the new path provided by the csv file. The csv file can now be reused to find the location of each processed image that we will use as input to our classifier.

4-2 Training

For this detection task, we randomly split the CheXpert dataset into training (222,914 images), validation (500 images) and test (234 images). The preprocessed training and validation images that have been resized and normalized are then fed into the network for training. During the training process, Adam optimizer with default β-parameters of $\beta 1 = 0.9$, $\beta 2 = 0.999$, learning rate 0.0001 and weight decay as 1e-5 was utilized. The loss function, Binary Cross Entropy with Logits, are accompanied to reduce a step of sigmoid function from Binary Cross Entropy Loss function for time efficiency. As a result of training, we created loss curves for training and validation data as shown in Figure 1.

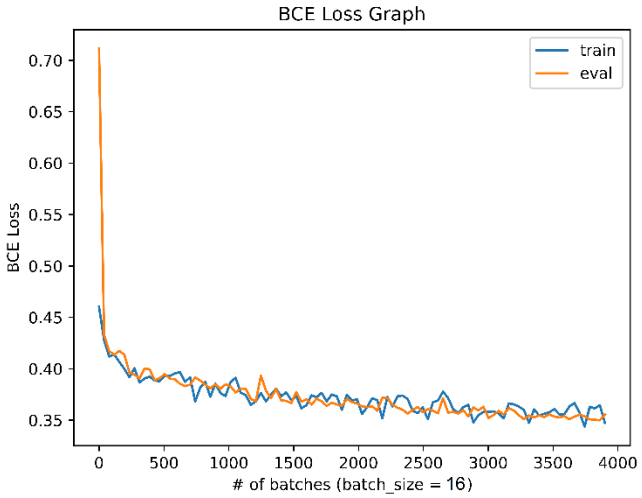

Figure 1. Binary Cross Entropy Loss curve

4-3 Testing

In the testing session, the two models learned through the training, DenseNet and ResNet, were used to test datasets to compare the prediction values by the model and labeled ground truth. Through this, False Positive Rate and True Positive Rate and Area Under the Curve (AUC) values for 14 diseases were generated through the sklearn package and plotted on ROC curves along with AUC measures.

5 Experimental Evaluation

We decided to try out our initial ideas and a few minor adjustments. As it turns out, our results were pretty good except three symptoms (Lung Lesion, Fractures, and Pleural Other) of abnormal patterns of the ROC curves. Excluding the abnormal symptoms, training results reached an average AUC score of 0.82. It is an unexpected success that we might only need to make a few minor adjustments to achieve better results. Figure 2 below shows our ROC and AUC for each diagnosis of the CheXpert dataset.

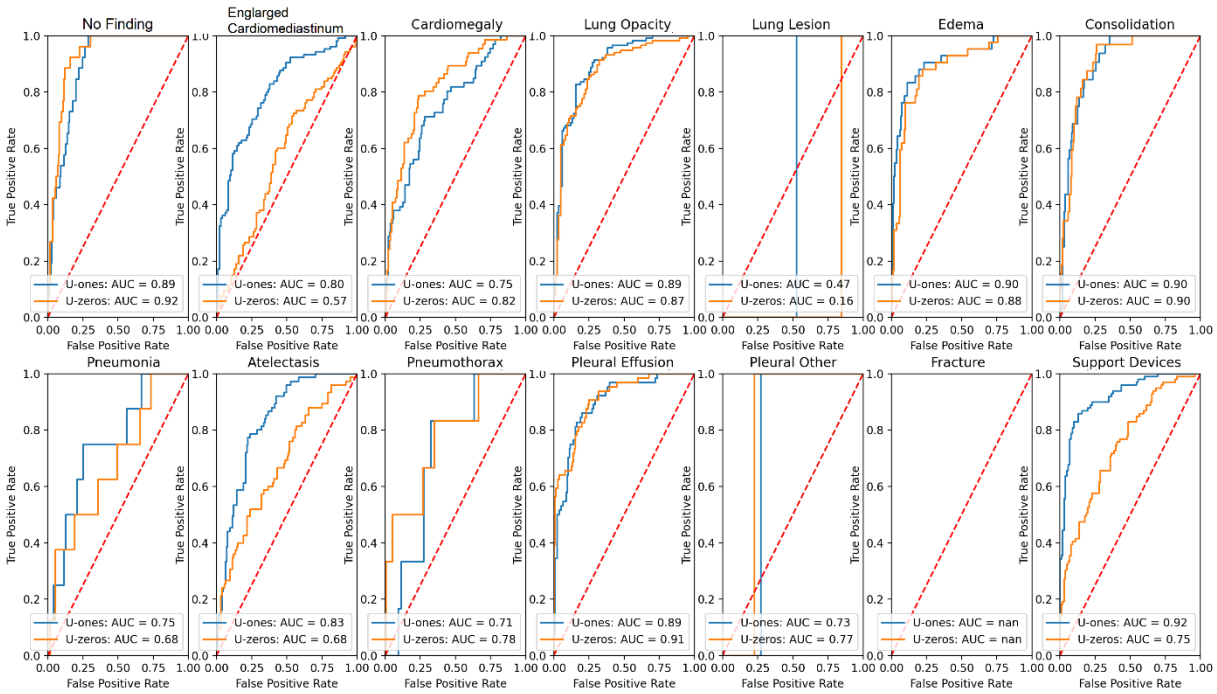

Figure 2. AUROC curves for 14 diseases detection

In Figure 2, there are certain diagnoses that did not perform as well as expected – for example our results in Lung Lesion, Fractures, and Pleural Other could have been better. To solve this, we investigated the images with the labels that are not performing well and attempted to see if other forms of preprocessing help predict those better. It turned out there is no positive case in the test data set for Fracture, and only 1 positive case exists for Pleural Other and Lung Lesion which most likely led to the poor results for these three diseases.

5-1 DenseNet vs. ResNet

In order to find a better fitted architecture for our problem, we experimented with two convolutional neural network architectures, specifically DenseNet and ResNet. These two architectures are models for image recognition, which have been recently attracting attention as winners of ImageNet and have been used widely with good performance. Both models have been developed for the purpose of enhancing information flow. The results of this experiment matched our expectations.

	Average	No Finding	Enlarged Cardiomediastinum	Cardiomegaly	Lung Opacity	Edema
ResNet	0.76	0.9	0.6	0.68	0.83	0.81
DenseNet	0.84	0.89	0.8	0.75	0.89	0.9
	Consolidation	Pneumonia	Atelectasis	Pneumothorax	Pleural Effusion	Support Devices
ResNet	0.88	0.77	0.78	0.6	0.79	0.67
DenseNet	0.9	0.75	0.83	0.71	0.89	0.92

Table 1. Average AUC for ResNet vs. DenseNet

As seen in Table 1 above, DenseNet's Average AUC was 0.84, which was 0.08 higher than ResNet's Average of AUC 0.76 in 11 symptoms excluding the 3 abnormal symptoms. The reason for this result originated from the differences on the structure of the two architectures' methods of maintaining the output from the previous layer. DenseNet, which stores all features from the previous layer with concatenate operation, had high accuracy, but consumed bigger memory and took more time than ResNet. In this regard, we decided to move the experiment with DenseNet architecture.

5-2 Uncertainty Label

When it comes to comparing the two uncertainty label approach methods, the average AUC score of U-Ones was 0.84, the U-Zeros was identified as 0.79 in 11 symptoms excluding the 3 abnormal symptoms. The U-Ones method had an average AUC score of 0.05 higher than that of U-Zeros. However, one interesting fact is that the overall score of U-Ones was not better in all 11 symptoms than U-Zeros. U-Ones were strong for six symptoms: Enlarged Cardiomediastinum, Lung Opacity, Edema, Pneumonia, Atelectasis and Support Devices, but other than Consolidation which scores the same for U-Ones and U-Zeros, the other 4 Symptoms of No Finding, Cardiomegaly, Pneumothorax and Pleural Effusion, U-Zeros was confirmed to be strong.

It is unclear the reason for the different performance of U-Zeros and U-Ones on ROC curves. However, we understand that the nature of symptoms might differ from one disease to the other, and to embrace variances of the data, we attempted to utilize the ensemble method of the two models but due to the time constraints this could not be completed to fruition.

5-3 Heatmap

We conducted an experiment to localize findings, the basis of symptom detections in X-rays. This experiment was implemented as follows. The weights for features from the trained model overlap with the frontal x-ray image of a specific patient, and then apply the colormap of the OpenCV package to darken the color of the estimated area to express findings prominently. Figure 3 represents an example of this heatmap performed on one patient. Of the 14 diseases, our model found Lung Opacity to be the most likely disease for this patient. Based on the labels from the data set this patient has the 3 symptoms Enlarged Cardiomediastinum, Edema, and Lung Opacity. This result of the heatmap model shows successful identification of diagnosed diseases.

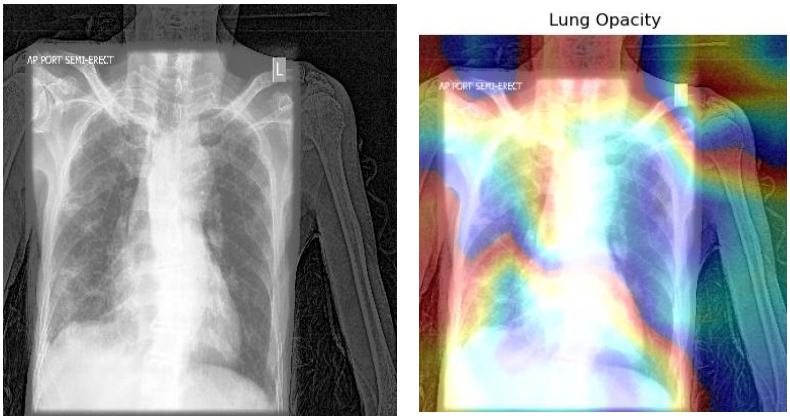

Figure 3 Heatmap visualization of findings in a radiograph.

6 Discussion

Our final results are not drastically different from previous experiments, but our preprocessing approach was beneficial to the outcome. We took a very simple approach and simply cropped the image to a standard size that is low quality. Even with the human eye, it's easy to get an intuition about what the diagnosis may be when comparing it to another diagnosis of the same classification. Reducing the amount of processing done on the image seems to be beneficial because of this. We also used the smaller CheXpert dataset which contains lower quality images. We believe these are sufficient for getting high accuracy results.

One of challenges we had is the time performance issue. It takes only one hour for image preprocessing, but about 2 days to train the model even though the GPU is used together. Because of this heavy work on the resource, it was necessary to execute under a thorough plan and difficult to make small tweaks and changes. The Ensemble model was also planned to be developed, but due to the issue, it was difficult to complete it within the project deadline.

Currently our preprocessing and training is slow, so it takes a long time to run and validate each new experiment. We have a few ideas that can help us iterate more quickly. One option is to crop the images to a smaller size of, 224x224, which is the minimum acceptable image size for DenseNet121 and run the same exact training as we're doing now to get a baseline. This will allow us to iterate and experiment quicker due to the decrease in training time required.

Once we're able to achieve better results on the limited data, we can start to scale back up to our original image size and hopefully retain the improvements from the smaller images. In terms of model improvement, we think that there are some improvements to be made with preprocessing. We will iterate through different cropping algorithms to alleviate some of the poor performance on certain labels and we will try different network topographies to see if we can achieve better results. One final area to help improve the results is to tune some of the parameters.

One other difficulty that we experienced was constructing the experimental environment to initiate the project. The computer specifications and the set up already installed on each team member's computer are different and requires significant effort to set up one common environment.

7 Conclusion

Through this analysis we addressed a real-world data science problem in healthcare and experienced providing end-to-end coverage of data science activities. We successfully conducted multiple experiments to detect 14 commonly spread diseases on chest X-rays using PyTorch and TensorFlow. We discovered that preprocessing the image and then feeding it through a trained model can provide increased benefits to the successful prediction of the 14 diseases.

References

1. J. Irvin, P. Rajpurkar, M. Ko, Y. Yu, S. Ciurea-Ilcus, C. Chute, H. Marklund, B. Haghgoo, R. Ball, K. Shpanskaya, J. Seekins, D. A. Mong, S. S. Halabi, J. K. Sandberg, R. Jones, D. B. Larson, C. P. Langlotz, B. N. Patel, M. P. Lungren, and A. Y. Ng. Chexpert: A large chest radiograph dataset with uncertainty labels and expert comparison. arXiv preprint arXiv:1901.07031, 2019.

2. P. Rajpurkar, J. Irvin, K. Zhu, B. Yang, H. Mehta, T. Duan, D. Ding, A. Bagul, C. Langlotz, K. Shpanskaya, M. P. Lungren, and A. Y. Ng. Chexnet: Radiologist-level pneumonia detection on chest x-rays with deep learning. arXiv preprint arXiv:1711.05225, 2017.

3. Q. Guan and Y. Huang. Multi-label chest x-ray image classification via category-wise residual attention learning. Pattern Recognition Letters, 2018.

4. Z. Li, C. Wang, M. Han, Y. Xue, W. Wei, L.-J. Li, and L. Fei-Fei. Thoracic disease identification and localization with limited supervision. CVPR, 2018.

5. Ge, Zongyuan, et al. "Chest X-Rays Classification a Multi-Label and Fine-Grained Problem." *Arvix*, Monash University, Melbourne, Australia. IBM Research Australia, Melbourne, 24 July 2018, arxiv.org/pdf/1807.07247.pdf.

6. Wang, Xiaosong, et al. "ChestX-ray8: Hospital-Scale Chest X-Ray Database and Benchmarks on Weakly-Supervised Classification and Localization of Common Thorax Diseases." *Arvix*, 1Department Of Radiology and Imaging Sciences, Clinical Center, 2 National Center for Biotechnology Information, National Library of Medicine, National Institutes of Health, Bethesda, MD 20892, 14 Dec. 2017, arxiv.org/pdf/1705.02315.pdf.

7. Mollura, Daniel J, Azene, Ezana M, Starikovsky, Anna, Thelwell, Aduke, Iosifescu, Sarah, Kimble, Cary, Polin, Ann, Garra, Brian S, DeStigter, Kristen K, Short, Brad, et al. White paper report of the rad-aid conference on international radiology for developing countries: identifying challenges, opportunities, and strategies for imaging services in the developing world. Journal of the American College of Radiology, 7(7):495–500, 2010.

8. Gao Huang, Zhuang Liu, Laurens van der Maaten, Kilian Q. Weinberger, et al. "Densely Connected Convolutional Networks" *Arvix*, preprint arXiv:1608.06993v5, 2018.

9. Kaiming He, Xiangyu Zhang, Shaoqing Ren, Jian Sun, et al. "Deep Residual Learning for Image Recognition" *Arvix*, Microsoft Research, 10 December 2015, arXiv:1512.03385v1.

Efforts Distributions / Individual Feedback

Member	Task Assigned	Feedback
Wang, Danliang	Research Lead Developer	My team is great. Everyone is passionate and making contributions!
Leventhal, Brandon K	Architect Lead Researcher Developer	Everyone did a great job and it was a pleasure to meet and work with the team.
Kim, Youjung	Project Management Researcher	In weekly meetings, everyone was participating and attentive on problems we had at moments. It was good experience for me and I learned a lot from this project.