Chest X-Ray (CXR) Disease Diagnosis with DenseNet

Doug Beatty, Filip Juristovski, Rushi Desai, Mohamed Abdelrazik
Georgia Institute of Technology, Atlanta, Georgia

Abstract

Chest X-ray imaging[1] is a crucial medical technology used by physicians to diagnose disease and monitor treatment outcomes. Training a human radiologist is a lengthy and costly process. Deep learning techniques combined with availability of larger data sets increases the feasibility of building automated models with performance approaching human radiologists.

We present a scalable deep learning model trained on the ChestXray14[7] data set of X-ray images to detect and correctly classify presence of 14 thoracic pathologies. We tried to beat current state of the art performance of models, and in some cases we were able to succeed. Class activation heatmaps are included which highlight areas of localization for the pathology in the image.

Introduction

A chest radiograph[1], or a chest X-ray (CXR) is one of the oldest and most common forms of medical imaging. A human radiologist requires significant training time and cost to be able to perform a comprehensive chest X-ray analysis with minimal error. Several types of abnormalities can arise in a chest radiograph that helps lead to detection and diagnosis of a multitude of diseases. With the vast number of different abnormalities and the overlapping reasons that might cause them, human error becomes a major contribution to poor diagnosis.

The revolution of machine learning and deep learning techniques combined with the availability of larger data sets[2] and big data processing systems[3] makes the analysis of X-ray images increasingly more realistic and the creation of automated models more feasible. The objective of this project is to train an efficient and scalable deep learning model based off of DenseNets[4], which can learn from a data set of X-ray images to detect and correctly classify 14 different pathology's. Automating the X-ray analysis makes the overall diagnosing process faster and less error-prone which significantly improves a patient's treatment procedure.

Approach

Our approach consists of 5 high-level activities:

1. Data acquisition
2. Image preprocessing - Apache Spark
3. Training DenseNet-121 deep learning model - Keras + PyTorch
4. Model validation and fine tuning
5. Model evaluation

The details of each of these activities are covered in subsequent sections.

Data acquisition

Two different datasets were considered for chest radiographs. The first is ChestXray14 from the NIH, the current results of this paper utilize the ChestXray14 dataset. The second is CheXpert, which provides a substantial improvement to ChestXray14 with more training images and labels. CheXpert was initially planned on being used to further improve performance, but due to rising costs of training the model, it was not pursued.

The full ChestXray14 dataset consists of 112,120 chest radiographs of 30,805 patients. The current model is trained off ChestXray14 and in some cases outperforms comparing models in performance for select pathologies.

CheXpert[2] consists of 224,316 chest radiographs of 65,240 patients. Each imaging study can pertain to one or more images, but most often are associated with two images: a frontal view and a lateral view. Images are provided with 14 labels derived from a natural language processing tool applied to the corresponding free-text radiology reports.

Data Information

ChestXRay14 has high resolution images which are not suitable as input to the model. Using a high resolution image significantly increases the number of input feature vectors increasing overall model complexity and training time. Another restriction to input image size is due to using a pretrained DenseNet model which was trained on ImageNet. Data set images were preprocessed before training using Apache Spark which is a scalable big data processing technology. The dataset was stored in Google Cloud Storage to provide a scalable mechanism for handling the large data set. Several down-sampling techniques were used to reduce image size.

A convolution neural network (CNN) is said to have an invariance property when it is capable to robustly classify objects even if its placed in different orientations. To enrich the input data set and increase the number of available training samples, horizontal flipping was applied randomly to images. This follows the DenseNet paper which found performance increases by adding horizontal flipping to the dataset.

Each input image is down sampled by resizing to 224x224 pixels. An input image generates one or more augmented versions of itself (e.g. by horizontal flipping). The DenseNet model utilizes transfer learning, and was originally trained on the ImageNet dataset, because of this the training data was normalized by the mean and standard deviation of the ImageNet dataset.

Some elementary statistics were gathered of the ChestXray14 dataset, this may be seen in Figure 1. Class imbalances become prominent when looking at this chart, Hernia only compromises of 0.28% of the total sample size, and Pneumonia only consists of 1.67%. Further data pre-processing could be done to address these class imbalances such as re-sampling the data set to get more even distributions and adjusting class weights when training the model.

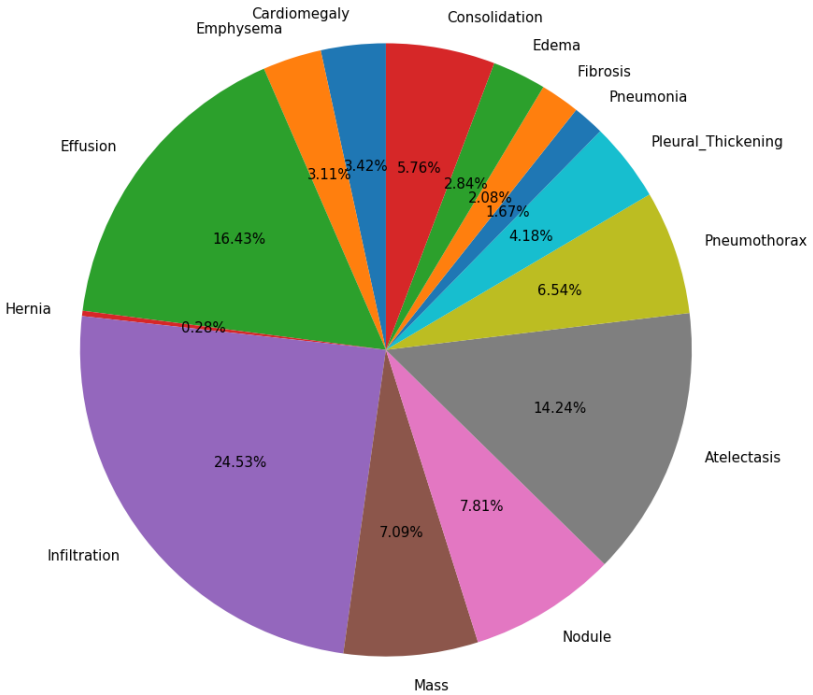


Figure 1: ChestXray14 Data Distribution

Method

Residual Networks (ResNets) allow us to train much deeper networks than a conventional CNN architecture since they handle the vanishing/exploding gradient problem much more effectively by allowing early layers to be directly connected to later ones. Dense Convolution Networks (DenseNets) are a form of residual network. Theoretically, it is expected that performance of models should increase as architecture grows deeper, but in reality as the network gets deeper, the optimizer finds it increasingly difficult to train the network due to the vanishing/exploding gradient problem. ResNet allow us to match the expected theoretical issue.

ResNets have significantly more parameters than conventional CNN networks. DenseNet retains all features of ResNet and goes further by eliminating some pitfalls of ResNet. DenseNets have much less parameters to train compared to ResNets (typically up to 3x less parameters). You may refer to Figure 2 which shows how DenseNet layers are connected. By concatenating all layer outputs together, the DenseNet helps solve the vanishing gradient problem as gradients no longer pass through arbitrarily deep convolutional towers. Instead each component is directly connected with other layers, which reduces overall parameters since redundant feature maps are not learned and better representational learning between layers occurs. One of the main insights between DenseNets and ResNets is that DenseNets concatenate features between layers, compared to ResNets which uses a summation. This dense connectivity pattern is the primary reason why less feature parameters are usually required for DenseNets.

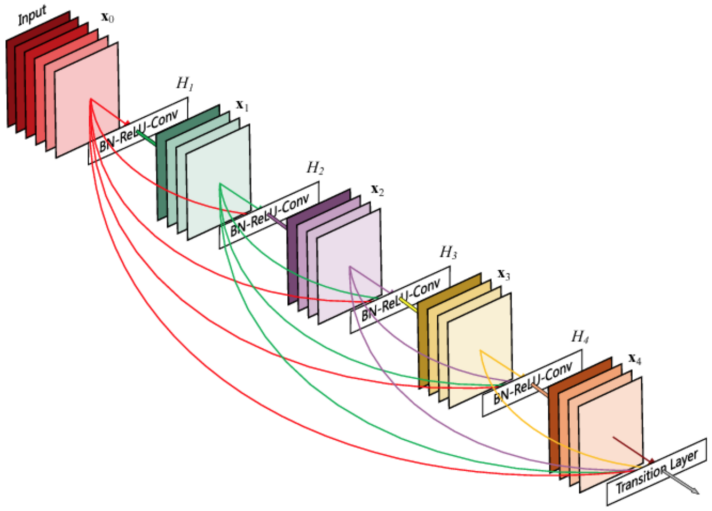


Figure 2: DenseNet Layers

The DenseNet models trained on ImageNet have a depth of 121. The architecture consists of a convolutional+pooling layer followed by 4 dense block stages. The dense block stages contain 6, 12, 24, and 16 units respectively, and each unit has 2 layers (composite layer with bottleneck layer). There is a single transition layer in between each dense block stage (for a total of 3 transition layers) which changes feature map size. Finally, there is 1 classification layer. 1 + 2(6) + 1 + 2(12) + 1 + 2(24) + 1 + 2(16) + 1 = 121. The base model we are using is DenseNet-121 BC with pre-trained weights from ImageNet. For feature extraction purposes, we load a network that doesn't include the classification layers at the top.

The model utilizes transfer learning due to a training size of only 112,120 samples, compared to the millions of samples which modern day models use. The base layers of a model are very generic to the data set, while later layers get more specific and tailored to their data set. Originally a few extra layers were added on top of the DenseNet model to help learn specifics of thoracic diseases, but due to over-fitting, these layers were removed. Normally in transfer learning, only the top few layers are re-trained since they are specific to their base training data set, but we found better results by re-training the entire model end to end to learn the ChestXray14 data set.

The machine learning pipeline consisted these primary stages.

1. Use Google Cloud Storage bucket as primary data access point for the data set. This provided fast and scalable usage which allows any data set to easily be used.
2. Pre-process and augment data set using Spark, downsampling, horizontal flipping, and mean/std normalization were some techniques utilized in this step.
3. Initial model development was done in Google Colab, then ported over to Amazon SageMaker for a scalable and persistent platform to train the model.

Metrics and Experimental Results

Accuracy, loss, and AUC scores are the main metrics used to evaluate the performance of the model.

Training and validation loss curves were used to provide insight into the model performance. An initial loss curve may be seen in in Figure 3. It may be seen that the model started over-fitting near the second epoch, because of this the training was stopped at only 8 epochs since further training would lead to no benefits. Based on the loss curves, adding more layers and increasing complexity would most likely lead to more over-fitting. Further investigation into better data pre-processing may help alleviate over-fitting, such as further data augmentation to increase sample size, better handling of class imbalances, or using more thorough data sets such as CheXpert. For class imbalances specifically, some of the classes such as pneumonia had extreme imbalances, 1872 cases out of a total of 112,120 images; only 1.67% of the overall data set.

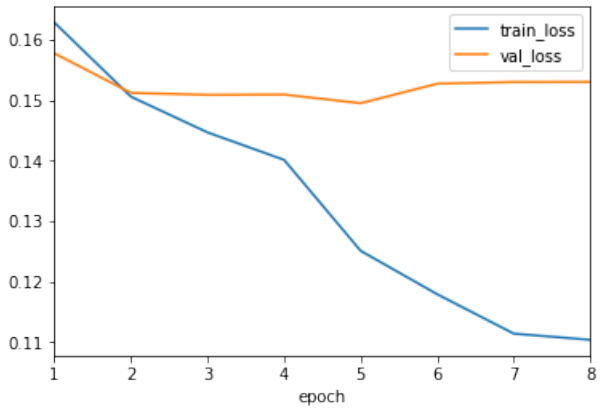


Figure 3: Model loss

The overall AUC score of the model compared to previous attempts may be viewed in Table 1 . Multiple variations of the model with different pre-processing were testing, the main 3 ones which provides promising results were stochastic gradient descent with momentum with no horizontal flipping, stochastic gradient descent with momentum and horizontal flipping, and Adam optimizer with horizontal flipping. Other variations of flipping and model hyper parameter tuning were attempted but they did not provide sufficient results and are thus not included.

Our variations of the model were able to get better performance in Atelectasis, Consolidation, Effusion, Fibrosis. Although these four diseases provided promising improvements, there were some large decreases AUC score compared to ChexNet. Infiltration, Nodule, and Pleural Thickening are three of the diseases whose AUC scores decreased substantially compared to CheXNet. Horizontal flipping actually decreased performance in most cases, we hypothesize this is due to applying this augmentation randomly across the entire data set, instead of focusing on specific classes, which may have exacerbated class imbalances which currently exist in the data set.

Pathology	Wang et al.[7]	Yao et al.[8]	Gündel et al.[9]	Liu et al.[6]	CheXNet[5]	Ours (SGD w/o flipping)	Ours (Adam w/ flipping)	Ours (SGD w/ flipping)
Atelectasis	0.716	0.772	0.767	0.781	0.8094	**0.8104**	0.7985	0.7799
Cardiomegaly	0.807	0.904	0.883	0.885	0.9248	0.8977	**0.9055**	0.8938
Consolidation	0.708	0.788	0.828	0.832	0.7901	**0.7961**	0.7945	0.7878
Edema	0.835	0.882	0.709	0.7	0.8878	0.8837	**0.8849**	0.8790
Effusion	0.784	0.859	0.821	0.815	0.8638	**0.8798**	0.8792	0.8658
Emphysema	0.815	0.829	0.758	0.765	0.9371	**0.9143**	0.8951	0.8334
Fibrosis	0.769	0.767	0.731	0.719	0.8047	**0.8284**	0.8063	0.7819
Hernia	0.767	0.914	0.846	0.866	0.9164	**0.9097**	0.8810	0.7738
Infiltration	0.609	0.695	0.745	0.743	0.7345	**0.6999**	0.6979	0.6854
Mass	0.706	0.792	0.835	0.842	0.8676	**0.8214**	0.8211	0.7837
Nodule	0.671	0.717	0.895	0.921	0.7802	**0.7506**	0.7226	0.7018
Pleural Thickening	0.708	0.765	0.818	0.835	0.8062	**0.7713**	0.7634	0.7505
Pneumonia	0.633	0.713	0.761	0.791	0.7680	**0.7678**	0.7498	0.7289
Pneumothorax	0.806	0.841	0.896	0.911	0.8887	**0.8674**	0.8533	0.8187

Table 1: AUC Scores Comparison

Discussion

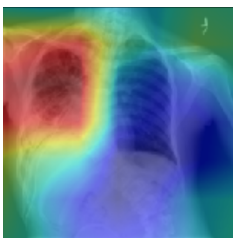
The initial goal of this paper was to reproduce the competitive results from the ChexNet paper, and then improve upon the performance by using a more substantial data set. Due to increasing costs of training the model on Amazon Sagemaker, the decision was made to not use CheXpert as originally planned.

Using only ChestXray14, better performance was achieved by using stochastic gradient descent (SGD) with momentum instead of the Adam optimizer. This was found to generalize better[12] and provide better results on the test set. Other improvements to the performance would be utilizing deeper versions of the DenseNet model such as DenseNet-169, and DenseNet-201. Utilizing these models in an ensemble pattern would also provide benefits to the overall performance.

Further plans to improve performance of the model were to integrate and train on the CheXpert dataset, this dataset has roughly twice the amount of images and also provides lateral chest X-rays, which have been found to account for 15% accuracy in diagnosis of select thoracic diseases[10]. The pre-processing spark code was developed to be agnostic to datasets and easily provide the necessary pre-processing on the data set.

Class Activation Maps (CAMS)[11] were generated to visualize where the model was focusing to make its classification, an example may be seen in 4. In the specific example generated by our model, the model correctly predicted Infiltration as the diagnosis and highlighted the right lung region which lead to the diagnosis. This is a useful tool for verifying correct and incorrect model predictions and help further fine-tune the model.

(a) Original Patient X-Ray



(b) Infiltration of right lung highlighted

Figure 4: Patient X-Ray & CAM Heatmap

Conclusion

DenseNets provide state of the art thoracic disease detection, at a fraction of the parameter cost of many modern day models. As a tool radiologists may use DenseNets to assist in initial diagnosis or verify patients diagnoses. ChestXray14 provides an excellent anonymoized dataset of chest X-rays which allows for the training of these high utility models. Using a DenseNet with data augmentation and hyperparameter tuning, we were able to surpass AUC and detection in select thoracic diseases. The use of class activation maps also enable verification of model focus and as a learning tool for radiologist to help identify what may lead to a diagnosis.

Moving onwards, CheXpert may be integrated as a larger data set and an ensemble model may be created using two convolutional towers, one for lateral photos and one for frontal photos. This combination of two separate orientations will help increase performance. Other variations of DenseNet may also be looked into such as the 169 and 201 layer versions of the model. The team is currently looking for funding to pursue such endeavours.

As chest X-rays are the most significant examination tool used in practice for screening and diagnosis of thoracic disease, the team hopes to provide better tooling and support for such a vital component of patient support. With limited radiologists available, approximately two thirds of the global population have deficient access to a specialist for screening and diagnosis[13]. Using this algorithm, patients without access to an expert may still be able to get expert level opinions and help reduce overall mortality rates throughout the world.

Challenges

We encountered the following challenges in this project:

1. Cost of training on public cloud service like Amazon SageMaker, especially GPU. We learned to focus more on getting good results on public resources before porting over to a cloud instance for training.
2. Format of Spark saving to and retrieving from HDFS, especially with png files.
3. Transferring File format between pre-proccessing step and model training step
4. Difficulties with getting the model to properly gain from transfer learning and the number of layers to unfreeze during training. Learned about varying depths of re-training layers and practices to ensure proper transfer learning such as normalization against the base data sets mean and std.
5. Difficult to implement unit tests within deep learning code. There were some bugs which affected training output, but through careful analysis they were discovered. More research into unit testing frameworks has been pursued.
6. Navigating a Dense121 Model to attach a hook to the correct layer. Needed to extract the correct neuron values and dimensions of the last convolution layer before flattening to generate the Class Activation Map (CAM).

Contributions

We had full participation and collaboration from all group members. We met frequently on Google Hangouts to discuss strategies and progress, about a dozen meetings in all. We also collaborated via Slack (over 1000 messages).

Filip Juristovski - modeling in Keras, model training/evaluation, prototype in Colaboratory, GitHub setup, Google Cloud Storage setup, paper

Rushi Desai - modeling in PyTorch/SageMaker, Model training/evaluation and billing monitoring, paper

Mohamed Abdelrazik - modeling in PyTorch/SageMaker, Class Activation Map (CAM), pre-processing in Spark, paper

Doug Beatty - EMR prototype, SageMaker Keras prototype, LaTeX formatting, presentation slides, paper

References

1. Siamak N. Nabili, M. (2019). Chest X-Ray Normal, Abnormal Views, and Interpretation. [online] eMedicine-Health.
2. CheXpert: A Large Dataset of Chest X-Rays and Competition for Automated Chest X-Ray Interpretation. [Internet]. Stanfordmlgroup.github.io. 2019.
3. FAN M, XU S. Massive medical image retrieval system based on Hadoop. Journal of Computer Applications. 2013;33(12):3345-3349.
4. Huang G, Liu Z, van der Maaten L, Weinberger K. Densely Connected Convolutional Networks [Internet]. arXiv.org. 2019.
5. Rajpurkar P, Irvin J, Zhu K, Yang B, Mehta H, Duan T et al. CheXNet: Radiologist-Level Pneumonia Detection on Chest X-Rays with Deep Learning [Internet]. arXiv.org. 2019.
6. Liu H, Wang L, Nan Y, Jin F, Pu J. SDFN: Segmentation-based Deep Fusion Network for Thoracic Disease Classification in Chest X-ray Images [Internet]. arXiv.org. 2019.
7. Wang X, Peng Y, Lu L, Lu Z, Bagheri M, Summers R. ChestX-Ray8: Hospital-Scale Chest X-Ray Database and Benchmarks on Weakly-Supervised Classification and Localization of Common Thorax Diseases. 2019.
8. Yao L. Weakly supervised medical diagnosis and localization from multiple resolutions [Internet]. Arxiv.org. 2019.
9. Guendel S, Grbic S, Georgescu B, Zhou K, Ritschl L, Meier A et al. Learning to recognize Abnormalities in Chest X-Rays with Location-Aware Dense Networks [Internet]. arXiv.org. 2019.
10. Raoof S, Feigin D, Sung A, Raoof S, Irugulpati L, Rosenow E. Interpretation of Plain Chest Roentgenogram. 2019.
11. Zhou B, Khosla A, Lapedriza A, Oliva A, Torralba A. Learning deep features for discriminative localization [Internet]. Arxiv.org. 2019.
12. Wilson A, Roelofs R, Stern M, Srebro N, Recht B. The Marginal Value of Adaptive Gradient Methods in Machine Learning [Internet]. Arxiv.org. 2017.
13. 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