Project Draft

Martin Saint-Jalmes[1], **Reynaldo Pena**[1]
[1]**Georgia Institute of Technology, Atlanta, GA, USA**

Abstract

This project aims to analyze radiographic lung images from the CheXpert Stanford hospital dataset to determine whether they have a lung disease and, if so, which disease they have. It relies on convolutional neural networks to learn from images, as well as big data tools to preprocess information. We explore several means to attempt to improve model AUC score, including data augmentation techniques and varying policies to handle uncertain and implicit data. The goal is to provide a supplement to health workers to better diagnose diseases. We provide our code and results.[2]

Introduction

Radiographic image disease classification is an important task in order to aid health workers better diagnose diseases and to save time doing so. The goal of this project is to create a lung radiographic image disease detection model to help doctors improve their disease diagnoses accuracy, especially when there is no clear consensus on the diagnosis. This is important because as many as five percent of U.S. adults receive incorrect diagnosis annually (Such et al. (2017)), and early correct diagnosis can save peoples' lives. Disease misdiagnosis has a huge cost for patients each year, ranging from misplaced and delayed treatments to death. A study conducted by the John Hopkins School of Medicine estimated that up to 80,000 deaths occur each year in the U.S. due to incorrect diagnoses (Newman-Toker et al. (2019)). Thus, this paper attempts to provide radiographers with another tool to improve correct lung disease diagnoses, intended to supplement their own diagnoses with advanced machine learning techniques. Improving the accuracy of the model means increasing the correct disease diagnoses rate, thus saving patients' lives. We use lung radiographic images from the Stanford Chexpert Dataset in RGB format, and then train convolutional neural network models to analyze these images. Our training data has fourteen response variables, and each variable has four possible outcomes: Positive (has disease), Uncertain, Negative, Implicit Negative (no *mention* of disease). Uncertain refers to cases where the doctor was not certain about the diagnosis. The main goal of this project is to find ways to best deal with the uncertain and implicit negative classifications in order to improve diagnoses, since this is where most improvement can be done. Our test set only presents Positive & Negative outcomes, but the distribution may be unbalanced so we use AUC as the means to evaluate our models.

Related Work

(Irvin et al. (2019)) describe the motivation for the release of the CheXpert dataset, one of the largest datasets of chest x-rays. Along with the dataset, the paper describes the labeling tool that was used to create the ground truth labels for this dataset (from freetext reports), an evaluation of the quality of labeling relying on expert appreciation (by several radiologists), and a baseline model for the task of predicting 14 diseases using uncertainty labels. (Pham et al. (2019)) present the state-of-the-art model that their team used to predict the 14 lung diseases and observations from the CheXpert dataset. Their model makes use of hierarchies, based on the interdependances between the diseases. The best AUC performance is achieved with a custom policy for uncertainty labels (label smoothing regularization) and an ensemble of CNNs. (Johnson et al. (2019)) release MIMIC-CXR-JPG, an alternative dataset (bigger than CheXpert), making MIMIC-CXR more accessible (in image JPGs, rather than DICOM format) to non-medical researchers. The labelling of the images from freetext reports was both performed by NegBio and the CheXpert labeling tool, and evaluated. The paper from (Ranjan et al. (2018)) describe the reasoning behind an alternative representation in the task of predicting diseases for the ChestX-ray14 dataset. They obtain better performance than other models at the time by using auto-encoders as preprocessing to retain information from the high-dimensional X-Ray images rather than downsampling the raw images as input for ImageNet (224x224). (Ge et al. (2018)) justify the use of bilinear pooling and a custom loss function (multi-label learning loss) to jointly learn a model that may take into account interactions between lung diseases on the ChestX-ray14 dataset. Their results show that such a method may improve overall AUC,

[2] `https://github.gatech.edu/msaintjalmes3/CheXpert`

and boost smaller architectures' performances. (Guan et al. (2018)) tackle the classification task from a different perspective than most papers, relying on spatial information in X-Rays to use an attention-learning technique. At the time of publication, their technique outperformed the state-of-the-art on the ChestX-ray14 dataset. The paper (Rubin et al. (2018)) offers some insight on using the relation between frontal and lateral X-Ray when training CNNs. Using their DualNet architecture, they were able to (most of the time) get better AUCs on the ChestX-ray14 dataset than if the frontal and lateral images had been used separately.

Approach & Metrics

Dataset

The dataset we used in this study is the CheXpert dataset from (Irvin et al. (2019)). The data consists of 224,316 chest radiographs of 65,240 patients. Some patients have multiple radiographs corresponding to side and front chest images. The chest radiographs were gathered from the Stanford hospital between 2002-2017. The images were labeled automatically from freetext radiology reports with a tool identifying mentions of 14 different diseases with a label of Positive, Negative, or Uncertain for each disease. Figure 1 shows the distribution of labels within CheXpert (we note they don't add to 100 as one patient may have multiple diseases). Finally, it is important to note that, as labels were generated (as opposed to human-picked), one can make a slight distinction between "explicit" and "implicit" negative labels: a radiology report may clearly state the absence of a certain disease ("explicit" negative), but not explicitly mention that every single of the other 14 diseases is absent. The policy regarding the handling of implicit negative and uncertain labels is a part of our experiment.

Pathology	Positive (%)	Uncertain (%)	Negative (%)
No Finding	16627 (8.86)	0 (0.0)	171014 (91.14)
Enlarged Cardiom.	9020 (4.81)	10148 (5.41)	168473 (89.78)
Cardiomegaly	23002 (12.26)	6597 (3.52)	158042 (84.23)
Lung Lesion	6856 (3.65)	1071 (0.57)	179714 (95.78)
Lung Opacity	92669 (49.39)	4341 (2.31)	90631 (48.3)
Edema	48905 (26.06)	11571 (6.17)	127165 (67.77)
Consolidation	12730 (6.78)	23976 (12.78)	150935 (80.44)
Pneumonia	4576 (2.44)	15658 (8.34)	167407 (89.22)
Atelectasis	29333 (15.63)	29377 (15.66)	128931 (68.71)
Pneumothorax	17313 (9.23)	2663 (1.42)	167665 (89.35)
Pleural Effusion	75696 (40.34)	9419 (5.02)	102526 (54.64)
Pleural Other	2441 (1.3)	1771 (0.94)	183429 (97.76)
Fracture	7270 (3.87)	484 (0.26)	179887 (95.87)
Support Devices	105831 (56.4)	898 (0.48)	80912 (43.12)

Figure 1: Data set label distribution (reproduced from (Irvin et al. (2019)))

For this work, we used the downsampled version of the CheXpert dataset where images are resized to a height of 320 pixels. This serves practical purposes, with the dataset being lighter (11 gigabytes instead of 439) and faster to process. The CNN architectures and models we considered using here all required input images to be resized to 224 by 224 pixels, so we wouldn't have been able to use the full potential of the original resolution images.

While there exists a way to evaluate a model on the CheXpert test set, we found the procedure somewhat impractical for evaluating our models rapidly. This is why we used the designated validation split as a stand-in test set, and built our own "internal" validation set to allow us to tune our CNN models. The matter of building this "internal" validation set is discussed in the following subsection.

Preprocessing

Our first preprocessing task consists of building an "internal" validation set for rapid evaluation purposes while fine-tuning a CNN model. Doing so allows us to avoid phenomena like overfitting on training data, so that we can pick an optimal model learned on the training set, but that generalizes well on unseen data (here, the "internal" validation set). Of course, such a process only has value if we change the training split so that it does not overlap with the "internal" validation set. In order to do this, we use the file provided in the CheXpert dataset that described the images from the original dataset, and split it into "internal" training and validation sets. This split is performed in a way such that no same patient can have radiographies both in the training and validation sets in order to reduce any potential biases (some diseases may carry over across multiple radiographies for the same patient). Considering the very big number of observations, we leveraged big data tools to execute this split. Using a Docker image [3] containing Hadoop and Spark, we load the description file into Hadoop Distributed File System (HDFS), and leverage Scala's `randomSplit` method to write two new descriptive files (containing the location of the radiographies and the labels' ground truths) for the "internal" training and validation set. Spark allows us to load the file in memory, perform the split in a distributed manner (using Resilient Distributed Datasets), and write the resulting new files in HDFS. In this same step, we can decide on the policy to experiment on. The specifics are mentioned in a further subsection (***Experiment design and policies***), but we use the same tools with Scala and Spark to replace blank values (corresponding to the "implicit" negatives) and "-1" (uncertain label) with certain constants. Likewise, the preprocessing is done in a distributed manner (with 4 workers), which allows us to process this data more efficiently.

Another preprocessing that leverages big data tools involves finding the per-channel mean and standard deviation of images. Normalizing the batch of images using these values generally yields better results. To find these values, we load the set of training images in HDFS, and use BigDL from (Dai et al. (2019)) to compute distributed representations of the pixels of the images. We can then find the mean and standard deviation values per pixels using Map-Reduce operations from Spark in Scala.

Further preprocessing steps are done directly using Pytorch. While this doesn't use big data tools, applying a series of preprocessing transformations with Pytorch dataloaders is an efficient means of performing data augmentation "on the fly", without creating new radiographies on the hard drive. Preprocessing steps such as resizing to 224 by 224 pixel images, and normalizing using the previously found values are applied to all data loaders. Conversely, data augmentation steps are only applied to the "internal" training data, not the "internal" validation, nor the "internal" test (i.e. real validation) sets. These data augmentation steps involve applying horizontal mirroring (diseases may affect both lungs the same way), color jitter, and light random cropping (which may crop out existing text on top of the radiographies) to ensure better model robustness.

Metrics

Considering our problem setting, we use Binary Cross-Entropy (BCE) as our loss function. This is for example what is also used in (Irvin et al. (2019)), (Pham et al. (2019)), (Guan et al. (2018)) and (Rubin et al. (2018)). The advantage of this loss function is that it allows for some great flexibility in experimenting with policies, as it performs a sigmoid activation. In this case, we're not constrained to use only 0 and 1 as values: (Pham et al. (2019)) use LSR to replace uncertain labels by uniformly sampled values that are close to 0 or 1. In this work, we initially set uncertain values to 0.66 and implicit negatives to 0.33 without needing to change the loss function.

As we are in an unbalanced setting (Figure 1), we use the area under the receiver operating characteristic (AUROC or AUC). For instance, a baseline model always predicting "No" would yield an expected 98.7% accuracy on "Pleural Other", so it would not be a suitable metric. We use AUC both to compare our results to other papers on the same validation set, and to perform an early stopping during our training phase by measuring AUC on our "internal" validation set. At each epoch, the AUC on the "internal" validation set is measured for each of the 14 diseases. We keep the model with the highest average AUC on the 14 diseases to counter the overfitting problem with too many epochs.

Experiment design and policies

As mentioned in (Irvin et al. (2019)), the fact that uncertain labels can be twice as numerous as positive labels makes the choice of a policy to deal with these labels an important one. On select diseases, a better policy can improve AUC by up to 5%, which can be critical in terms of model adequacy. In this work, we introduce an additional choice in policy regarding "implicit" negatives, corresponding to blank values in the dataset. While in all likelihood, the absence of mention of a disease in a report could be considered to be a negative label for this disease, we wanted to encode stronger priors on "explicit" negatives than implicit ones. We therefore experimented in settings where an "implicit" negative was assigned a label of 0 (i.e. same policy as in other works), or 0.33 (tending to 0, but not ruling out an omission in written reports). Likewise, we design different policies regarding the processing of uncertain values (e.g. set them to 1, 0 or 0.66).

[3] http://www.sunlab.org/teaching/cse6250/spring2020/env/env-docker-compose.html

Our experiments were run with two different types of architectures, corresponding to some of the state-of-the-art architectures for object recognition: DenseNet-161 from (Huang et al. (2017)) and ResNet-152 from (He et al. (2017)). These models were adapted so that the final layer only outputs 14 values instead of 1000, and we introduced a sigmoid function to retain values in the $[0, 1]$ range. In both cases, we ran experiments training from scratch, or finetuning from pretrained weights (learned from the ImageNet classification task), freezing the lower layers of the networks. All models were run for 5 epochs, with an Adam optimizer using default values $\beta_1 = 0.9, \beta_2 = 0.999$. The learning rate for models trained from scratch was 10^{-3}, but was lowered to 10^{-4} for finetuning the pretrained models.

Experimental Results

We ran part of our desired experiments on remote servers rented on Google Cloud Platform (GCP), using an NVIDIA T4 GPU with 16GB of VRAM for about six hours per model. Due to the GPU's memory limits, the batch sizes were constrained to 48 and 64 (respectively for DenseNet-161 and ResNet-152 models) trained from scratch, and 512 for finetuning models.

	ResNet-152			DenseNet-161			
	scratch implicit=0.33 uncertain=0.66	scratch implicit=0 uncertain=0.66	pretrained implicit=0 uncertain=0.66	scratch implicit=0 uncertain=0.66	pretrained implicit=0 uncertain=0.66	pretrained implicit=0 uncertain=0	pretrained implicit=0 uncertain=0
No Finding	0.773	0.829	0.810	0.786	**0.834**	0.794	0.834
Enlarged Cardiomediastinum	**0.824**	0.460	0.469	0.334	0.512	0.526	0.552
Cardiomegaly	0.752	0.730	0.720	0.757	0.733	**0.803**	0.746
Lung Opacity	0.810	**0.870**	0.809	0.843	0.823	0.859	0.824
Lung Lesion	0.000	0.004	0.006	0.012	0.060	0.015	0.062
Edema	0.804	0.815	**0.824**	0.795	0.815	0.791	0.797
Consolidation	**0.822**	0.792	0.769	0.759	0.817	0.752	0.742
Pneumonia	0.513	0.391	0.565	0.259	**0.580**	0.228	0.460
Atelectasis	**0.778**	0.753	0.765	0.777	0.762	0.735	0.722
Pneumothorax	0.347	0.498	**0.640**	0.447	0.636	0.477	0.621
Pleural Effusion	0.799	**0.860**	0.787	0.858	0.820	0.855	0.819
Pleural Other	0.089	0.109	0.241	0.078	0.278	0.087	**0.302**
Fracture	0.000	0.000	0.000	0.000	0.000	0.000	0.000
Support Devices	0.694	0.741	0.738	0.753	0.721	**0.794**	0.720
Overall Mean AUC	0.572	0.561	0.582	0.533	**0.599**	0.551	0.586
5-observation focus Mean AUC	**0.791**	0.790	0.773	0.789	0.789	0.787	0.765

Table 1: AUC scores on the validation set (i.e. "internal" test set), and averages (focus is on the 5 in bold-italics).

Discussion

The results on the validation set near the baseline (AUC of 0.811) established in (Irvin et al. (2019)) with the *U-Zeros* (implicit=0, uncertain=0) policy on the 5 observations of particular interest (Cardiomegaly, Edema, Consolidation, Atelectasis, Pleural Effusion). Our model based on ResNet-152 trained from scratch and assigning a non-zero label to "implicit" negative cases shows to be competitive with other well-established policies (e.g. DenseNet with *U-Zeros*). As this leading policy has not been fully explored yet (with DenseNet-161, or on pretrained models), we have high hopes that it may prove useful for some diseases. We note that we retrieve the same observations from previous papers, where no one policy dominates for all disease types. Finally, we wish to investigate the strange results from the "Fracture" class, which may come from an unfortunate "internal" training-validation split, failing to help generalize.

Conclusion & Possible optimizations

Correct disease diagnoses is top priority for healthcare professionals. This project sought to create more accurate lung disease diagnosis using radiographic images by applying different preprocessing steps and exploring new policies to handle uncertain and implicit output variables. While our best model was competitive with the original paper's baseline, it fails to reach state of the art performance of the more complex, hierarchical models. We will seek to improve models by training for more epochs (learning curves suggest we are not overfitting yet at the end of the 5 epochs). We are also seeking to experiment with the U-ones and U-Multiclass policy (modeling explicitly the 4 classes instead of bringing them back to the two Positive/Negative) and the DualNet model (learning separate models for frontal and lateral images), as they have shown promising results in their respective papers.

References

1. Such MV, Lohr R, Beckman T, et al. Extent of diagnostic agreement among medical referrals. Journal of Evaluation in Clinical Practice 2017;23:870?4. doi:10.1111/jep.12747
2. Newman-Toker DE, Schaffer AC, Yu-Moe CW, et al. Serious misdiagnosis-related harms in malpractice claims: The “Big Three” - vascular events, infections, and cancers. Diagnosis 2019;6:227?40. doi:10.1515/dx-2019-0019
3. Irvin J, Rajpurkar P, Ko M, et al. CheXpert: A Large Chest Radiograph Dataset with Uncertainty Labels and Expert Comparison. Proceedings of the AAAI Conference on Artificial Intelligence 2019;33:590?7. doi:10.1609/aaai.v33i01.3301590
4. Pham, Hieu H., Le, Tung T., Dat, Tran Q., et al. Interpreting chest X-rays via CNNs that exploit disease dependencies and uncertainty labels. Neurocomputing (preprint) 2019; arXiv:1911.06475v2
5. Johnson, Alistair E. W., et al. MIMIC-CXR, a De-Identified Publicly Available Database of Chest Radiographs with Free-Text Reports. Scientific Data, vol. 6, no. 1, 2019, doi:10.1038/s41597-019-0322-0.
6. Ranjan, Ekagra, et al. Jointly Learning Convolutional Representations to Compress Radiological Images and Classify Thoracic Diseases in the Compressed Domain Proceedings of the Twelfth Indian Conference on Computer Vision, Graphics and Image Processing, 2018, doi:10.1145/3293353.3293408
7. Ge, Zongyuan, et al. Chest X-rays Classication a Multi-Label and Fine-Grained Problem, 2018, arXiv:1807.07247v3
8. Guan, Qingji, and Yaping Huang. Multi-Label Chest X-Ray Image Classification via Category-Wise Residual Attention Learning. Pattern Recognition Letters, vol. 130, 2020, pp. 259?266., doi:10.1016/j.patrec.2018.10.027.
9. Rubin, Jonathan, et al. Large Scale Automated Reading of Frontal and Lateral Chest X-Rays using Dual Convolutional Neural Networks, 2018, arXiv:1804.07839
10. Yasaka K, Abe O. Deep learning and artificial intelligence in radiology: Current applications and future directions. PLOS Medicine 2018;15. doi:10.1371/journal.pmed.1002707
11. Dai JJ, Li Z, Wang J, et al. BigDL. Proceedings of the ACM Symposium on Cloud Computing - SoCC 19 Published Online First: 2019. doi:10.1145/3357223.3362707
12. Huang G, Liu Z, Maaten LVD, et al. Densely Connected Convolutional Networks. 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR) Published Online First: 2017. doi:10.1109/cvpr.2017.243
13. He K, Zhang X, Ren S, et al. Deep Residual Learning for Image Recognition. 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR) Published Online First: 2016. doi:10.1109/cvpr.2016.90