Weakly supervised ML model to detect anomalies in chest radiographs

Shruthi Pasumarthi
Graduate Student
Electrical and Computer Engineering Department
UNC Charlotte
spasuma5@uncc.edu

***Abstract*—Each year, due to misdiagnosis, a lot of people in the world are affected, which may result in fatalities. To avoid this, a lot of people suggest to get 'second opinions', which costs a lot of money as they are paying for the same thing, but from a different doctor. A lot of people cannot afford this sometimes due to which they live in fear as they suspect if the doctor has given them the right diagnosis or not.**

Having a 'check' on the diagnosis given by doctor, if very helpful, but due to the lack of workforce in the medical field, this is not very possible. Hence by using Machine Learning algorithms, the device can give the doctor an idea if the diagnosis is normal or abnormal, based on the radiologist's notes.

***Index Terms*—Labeling functions, Weak Supervision, Machine Learning**

I. Introduction

Machine learning models have come a long way from being rudimentary and unreliable to a field where there is so much progress and research. people and corporations are depending on Machine Learning so much, that it has replaced employees in some tasks, helping the company in cutting costs.

Due to the high demand of Machine Learning and its applications in different fields, there is so much research in this field, from making an accurate model with high complexity which requires a lot of resources, to more light-weight models, which can be used for personal uses also.

In this project, I experimented with weak supervised models used in the medical field, where I implemented text-classification. Given the notes written by the radiologists, the model can classify the corresponding radiograph as normal or abnormal. Based on this, the doctor can give a diagnosis to the patient.

II. DataSet

I used the data from the Openi website, which is a service provided by the National Library of Medicine [2]. The chest radiographs that I used is provided by Indiana University hospital network. The images are provided in two formats, in a PNG and a DICOM format. DICOM images are used very frequently in the medical industry. However, to use it for Machine Learning applications, it must be converted into the PNG or JPEG or JPG format.

The same dataset is also available in Kaggle [4], with the csv files of the notes of the chest radiographs made by the radiographer.

There are a total of 7470 images however, only 3,955 have the corresponsing radiology reports. The dataset has a combination of frontal and a side view xray of the chest and is divided into three parts, training, development (dev), and the test datasets. The development dataset helps us to tune the hyperparameters as per the results received after training and running on the development dataset. The development set is more commonly known as the validation dataset.

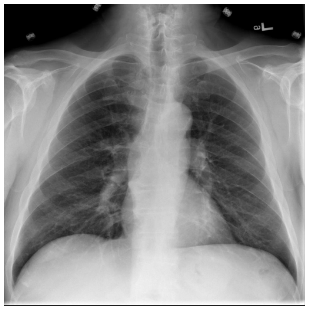

Fig. 1. Frontal xray

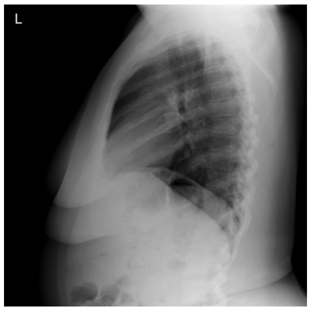

Fig. 2. Side xray

The training dataset 2630 images, out of which there are 63.8% abnormal images. The development dataset has 376 images, out of which there are 63% abnormal images and in the test dataset, there are 378 images out of which there are 61.6% abnormal images.

```
2630 train examples: 63.8% Abnormal
376 dev examples: 63.0% Abnormal
378 test examples: 61.6% Abnormal
```

Out1: Percentage of abnormalities in each dataset

The dataset also has a excel sheet accompanying it with details about the image such as the radiographer's notes of the xray, a label for supervised learning, the path of the corresponding image and a few keywords describing the xray. There are 3 excel sheets with this information, one for the training dataset, development/validation dataset and the testing dataset.

```
RAW TEXT:

COMPARISON: Chest x-XXXX XXXX INDICATION: XXXX in bathtub FINDINGS: The lungs and pleural spaces show no acute abnormality. Hyperexpanded lungs. Calcified right upper lobe granuloma, unchanged. Heart size and pulmonary vascularity within normal limits. No displaced rib fractures. IMPRESSION: 1. Hyperexpansion without acute pulmonary abnormality.

IMAGE PATHS:

./data/openi/xrays/CXR2824_IM-1245-13001.png

LABEL: 1
```

Out 2: Output of the excel sheet when printed in Jupyter Notebooks

III. Installation

For the project and all its dependencies to run, I had to install a docker as per the tutorial in [5]. For the visualization, general libraries such as seaborn or matplotlib was not used. I instead had to clone the metal repository [6]. After cloning the repository, I was getting a lot of errors. I then changed the entire repository by putting the the path for each dependant .py file which then gave me the expected results.

IV. Methodology

This project also uses an auxiliary package called Snorkel MeTal [6] for the visualization of the results. It first sets up the environment by loading all the necessary packages and setting the path.
Then the percentage of abnormal labels are calculated, by reading the 'label', 'xray_path' and the 'text' columns. The results are printed out. Also, the radiographer notes which is considered as raw text and the label associated with it is also printed with the path of the image.

After this the labeling function is called upon. In the labelling function, there are many parameters that has to be met before classifying the xray as a normal or an abnormal one. There are three ways of classifying and labeling the data, one is called a 'General pattern labeling function', where based on a few words, the radiographs are classified into normal or abnormal. For this the accuracy of the labelling function is 78%. For the second type of classification, it only classifies any case of lung hyperdistention as an abnormal radiograph. For this, the accuracy received is 44.6%. The last type of classification is the Structural classification, where it bases its decision on the number of characters in the radiographer's notes is less than 280. The accuracy received for this is 63%. However, even with the least accuracy, the lung hyperdistention type is used as it actually detects all the cases of normality and abnormality, whereas the other two classifications cannot detect as well as the lung hyperdistention type of classification.

However, this technique is not used very often, as this can only be used for text. There are many other factors that go into deciding the label for a particular piece of text. This is done by the 'labeling_function.py' file, which has a lot of parameters which decide the label of the text. This labeling function has a lot of parameter definitions based on all of which the text is classified. In this project there are 18 such parameters based on which the label, normal or abnormal is put on a text.

To then combine all the all the individual parameters, a hyperparameter search is done on the data. The hyperparameter search runs through the all the labeling functions and their individual weights and picks the best combination of weights that gives the best results. The hyperparameter search is evaluated on the developmental set giving an accuracy of 86.2%. The Region under the curve score is also calculated which is 92%. This score is considered throughout the project as it defines the separability of the data, which is important in classification.

This hyperparameter search model is now used to create weak supervised generative labels. The labeling gives a probability of each label. We can binarize the label, which also gives good results, implying that our model is well-chosen. The gold labels in the histogram are the ground-truth labels.

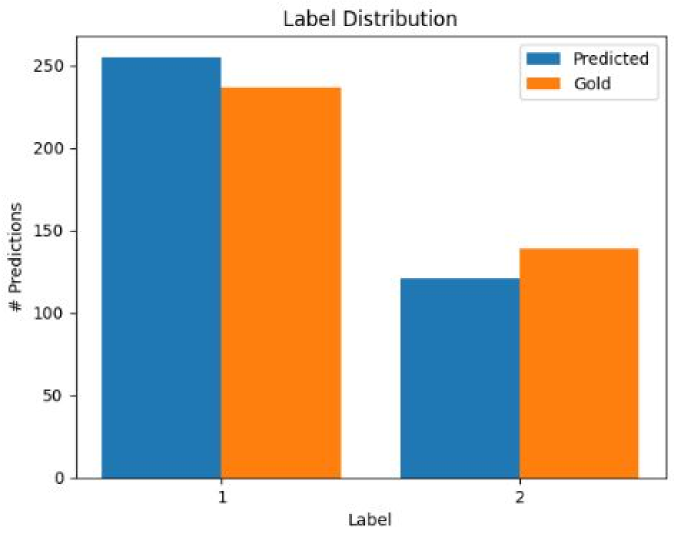

Fig. 3. Histogram of the distribution of the predicted and gold labels

Now with the generated weak labels, we train our weakly supervised model. Here I have chosen to make a comparison

of the performance of different models, all belonging to the ResNet family. I have chosen the ResNet18, ResNet34, ResNet50 and the ResNet 101 to compare on.
I compare the weak supervised models to fully supervised models, also using the same ResNet networks, Resnet18, ResNet34, ResNet50, and ResNet101.
I trained all the models, the fully supervised and the weak supervised models for 5 epochs each.

V. Results

The training took a lot of time on my personal computer as I do not have a GPU. For the heaviest model, the ResNet101, it took me almost 3-4 hours to train for 5 epochs. To get better accuracy, I tried increasing the number of epochs, however after 5 epochs, the accuracy started reducing, which led me to keep the number of epochs as 5. The ROC-AUC scores are also calculated and considered as they give the degree of separability (as mentioned before).

Weakly Supervised Models:

Network	Accuracy	ROC-AUC
ResNet18	67.8%	69.4%
ResNet34	67.8%	66.7%
ResNet50	66.5%	64.7%
ResNet101	63%	54.8%

Fully Supervised Models:

Network	Accuracy	ROC-AUC
ResNet18	69.9%	74.8%
ResNet34	74.7%	71.6%
ResNet50	67.8%	67.3%
ResNet101	70.5%	62.3%

The ROC-AUC scores are received when running the model on the test data.

From both the tables, we can conclude the following:

- For Weakly supervised models, Resnet18 and Resnet34 gave the best results with accuracy of 68%.
- For fully supervised models, ResNet34 gave the highest accuracy of 75%
- The highest ROC-AUC score for both fully and weakly supervised model was obtained using the ResNet18 (74.8%, 69.4%)

VI. Future Works

I would like to expand the types of networks I use. I also want to try and implement this on more lightweight models such as the MobileNet models. I tried to implement this on other networks other than the ResNet family, such as the VGG16 netowrk and the MobileNet V2, however I got a lot of errors due to the different architectures of both the models.

I would also like to extend this project to the detection of more specific diseases such as pneumonia.

Acknowledgment

I would like to thank my professor, Dr. Minhaj Nur Alam, for helping me out with the project idea and guiding me throughout the implementation of this paper.

With this project, I have gone into a field that was not taught, but I was able to show some results, all thanks to my professor for his guidance.

References

[1] Jared A. Dunnmon et. al. , “Cross-Modal Data Programming Enables Rapid Medical Machine Learning” arXiv:1903.11101v1 [cs.LG], March 2019

[2] Dina Demner-Fushman et. al. , ”Preparing a collection of radiology examinations for distribution and retrieval”, 10.1093/jamia/ocv080. Epub 2015 Jul 1.

[3] Sala, Fred. ”Auto LF generation: Lots of little models, big benefits”. SnorkelAI. May 31 2022. https://snorkel.ai/generating-labeling-functions-lfs-automatically/

[4] Raddar, “Chest X-rays (Indiana University)” . Kaggle. 2019. https://www.kaggle.com/datasets/raddar/chest-xrays-indiana-university

[5] Chen, Vincent, “Snorkel Team” GitHub. October 31 2022. https://github.com/snorkel-team/snorkel

[6] Bach, Stephen “HazyResearch” GitHub. September 4 2019. https://github.com/HazyResearch/metal