Chest X-ray Disease Diagnosis

Lin Shao, Tong Wu, Yahui Ke, Yinuo Liu

Abstract

We developed a web application that can diagnose diseases based on Chest X-ray. Users can upload Chest X-ray images, and our backend algorithms will provide the five most possible diseases with probabilities. The backend model is a deep convolutional neural network (VGG16) trained on ChestX-ray14 data set. The web application used use Django web framework which is hosted in AWS.

Introduction

Chest X-ray is the most popular method to diagnose related diseases, and early correct diagnosis and treatment is critical to save patients’ lives and improve health care system efficiency [1]. However, accurate detections highly depend on the radiologist’s expertise, which requires long time training and practicing. With the advancements in deep learning over the past decade, we can train computers to detect various objects or classify different groups with high accuracy. In this project, we implement two CNN architectures – DenseNet121 [2] and VGG16 to detect 14 diseases from the ChestX-ray14 dataset. We build a web application which applies the best model and hyper parameters at backend and provides potential user a user-friendly tool.

Literature Review

Recently, multiple deep neural networks [2-4, 7, 8] have been proposed to detect pathologic patterns from chest x-rays since a large-scale data set [3] from National Institutes of Health Clinical Center became available. Wang et al. [3] applied a unified Deep Convolutional Neural Network (DCNN) based on four pre-trained models, AlexNet, GoogleNet, VGGNet-16 and ResNet-50. This work is a performance benchmark and triggered several other groups to study the application of CNNS in disease detection. Yao et al. [4] utilized a combination of CNNs and Long Short-Term Memory Networks (LSTM) to predict the 14 thoracic diseases and explore the interdependencies among them. Rajpurkar et al. [2] presented a modified Densely Connected Convolutional Networks (DenseNet) to classify the 14 diseases. The combination of dense connections [5] and batch normalizations [6] raised the Area Under Curve (AUC) scores after fine-tuning. Li et al. [7] used a pre-trained residual neural network (ResNet) to extract features and deployed a CNN to produce a disease probability map. Guendel et al. [8] proposed a location aware Dense Networks (DNetLoc), which achieved better AUC scores on multilabel classification and pathology location.

**Dataset**

In this study, we use ChestX-ray dataset, which was extracted from the clinical PACS database at National Institutes of Health Clinical Center by Wang and et al [3]. It includes 112,120 frontal-view X-ray images of 30,805 patients with 14 disease image labels.

One X-ray image can be labeled with no disease, one disease or multiple diseases. The distribution of X-ray images labelled with different number of diseases is in table 1. About 54% images indicate of no disease, and 28% images have only one disease. There are only 0.1% images are labelled as more than 5 diseases.

Table 1. X-ray Image Distribution with Different Number of Diseases

|  |  |  |
| --- | --- | --- |
| **Number of Disease on One Image** | **Count of X-Ray images** | **Percentage of X-Ray images** |
| 0 | 60,361 | 53.8% |
| 1 | 30,963 | 27.6% |
| 2 | 14,306 | 12.8% |
| 3 | 4,856 | 4.3% |
| 4 | 1,247 | 1.1% |
| 5 | 301 | 0.3% |
| ≥ 5 | 86 | 0.1% |
| **total** | **112,120** | **100.0%** |

Among all the X-ray images, the frequency of each disease is listed in table 2. As we can see, Infiltration occurs the most frequently with the percentage of 17.7%, while Hernia has the least frequency of only 0.2%. To account for the imbalanced data, the class weights for the training data were passed to the model.

Table 2. Disease Frequency in X-ray Images

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Disease** | **Frequency** |  | **Disease** | **Frequency** |
| Infiltration | 17.7% |  | Pleural Thickening | 3.0% |
| Effusion | 11.9% |  | Cardiomegaly | 2.5% |
| Atelectasis | 10.3% |  | Emphysema | 2.2% |
| Nodule | 5.6% |  | Edema | 2.1% |
| Mass | 5.2% |  | Fibrosis | 1.5% |
| Pneumothorax | 4.7% |  | Pneumonia | 1.3% |
| Consolidation | 4.2% |  | Hernia | 0.2% |

We split the whole dataset into training, validation and testing by 70%, 10% and 20% based on patient ID rather than image names. The reason is that, we need make sure the X-Ray images of the same patient do not cross dataset. The resolution is 10241024 for all images. For typical CNN classification, image augmentation is necessary to cover the variations in actual prediction. However, Chest X-ray images are quite similar, we do not expect to see different rotation, lighting condition, or noises. Based on Rajpurkar et al. [2], we compared random horizontal flip with no augmentation in training and observed no performance difference. The input for our model is RGB image, and outputs are 14 class with probabilities.

**CNN Model Architecture**

Our CNN models were implemented using Keras with TensorFlow as backend. We used two models, DenseNet121 and VGG16. These two models have shown impressive results in image recognition [9, 10]. VGG16 have 16 convolutional layers with small receptive fields (3 x 3). Every 2 to 3 convolutional layers are connected by a pooling layer. DenseNet121 has 4 dense blocks. In each dense block, the layers have the same dimensions of the feature maps while the number of filters changes. Both models are obtained from Keras applications. We added a fully connected layer at the end to predict for 14 classes. The summarized model architectures are shown in the table 3 and table 4.

Table 3. Architecture of DenseNet121

|  |  |  |
| --- | --- | --- |
| **Layer** | **Output Shape** | **Number of Parameters** |
| Input | (n, 224, 224, 3) | 0 |
| DenseNet121 | (n, 1024) | 7,037,504 |
| Output | (n, 14) | 14,350 |

Table 4. Architecture of VGG16

|  |  |  |
| --- | --- | --- |
| **Layer** | **Output Shape** | **Number of Parameters** |
| Input | (n, 224, 224, 3) | 0 |
| VGG16 | (n, 512) | 14,714,688 |
| Output | (n, 14) | 7,182 |

We used binary cross entropy as the loss function. One single binary cross entropy loss is:

Since the disease labels are unbalanced, the class-weighted loss is:

With 14 diseases for prediction, our final loss in Keras is:

Training Results

After finely tuning the hyper parameters, we ended up with the model implementation details as Table 5.

The AUC scores on the test sets using the two above VGG16 and DenseNet 121 model are listed in Table 6. We can see that the two models didn’t show significant difference in performance. We also compared them with the AUC scores from Wang [3], Yao [4] and Rajpurkar [2] papers. Wang used ResNet-50, and Yao used DenseNet. Our performance is slightly better than theirs. Rajpurkar used DenseNet model. Our performance is slightly worse. The possible reasons for the performance difference are image pre-process and hyper parameters.

Table 5. Training Hyper Parameters

|  |  |
| --- | --- |
| **Data Generator** | 1) resize image from 1024 1024 to 224 224;  2) random horizontal flip (optional);  3) standardize based on ImageNet. |
| **Batch Size** | 32 |
| **GPU** | 16G memory |
| **Optimizer** | Adam |
| **Learning Rate** | 0.0001 for VGG16; 0.001 for DenseNet 121 |
| **Learning Rate Reduced Factor** | 0.2 |
| **Callback Function** | Calculate mean AUC at each epoch end |

Table 6. Performance Comparison (AUC)

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Disease** | **VGG16** | **DenseNet121** | **Wang [3]** | **Yao [4]** | **Rajpurkar [2]** |
| Atelectasis | 0.810 | 0.808 | 0.716 | 0.772 | 0.809 |
| Cardiomegaly | 0.883 | 0.895 | 0.807 | 0.904 | 0.925 |
| Consolidation | 0.791 | 0.793 | 0.708 | 0.788 | 0.790 |
| Edema | 0.892 | 0.877 | 0.835 | 0.882 | 0.888 |
| Effusion | 0.872 | 0.875 | 0.784 | 0.859 | 0.864 |
| Emphysema | 0.886 | 0.898 | 0.815 | 0.829 | 0.937 |
| Fibrosis | 0.776 | 0.790 | 0.769 | 0.767 | 0.805 |
| Hernia | 0.842 | 0.863 | 0.767 | 0.914 | 0.916 |
| Infiltration | 0.706 | 0.706 | 0.609 | 0.695 | 0.735 |
| Mass | 0.839 | 0.84 | 0.706 | 0.792 | 0.868 |
| Nodule | 0.751 | 0.739 | 0.671 | 0.717 | 0.78 |
| Pleural Thickening | 0.784 | 0.793 | 0.708 | 0.765 | 0.806 |
| Pneumonia | 0.737 | 0.759 | 0.633 | 0.713 | 0.768 |
| Pneumothorax | 0.861 | 0.867 | 0.806 | 0.841 | 0.889 |
| **Mean AUC** | **0.816** | **0.822** | **0.738** | **0.803** | **0.841** |

In the training, DenseNet121 consumed 215 ms, while VGG16 consumed 155 ms on one 32 images batch on Tesla V100, saving 39% time. In our model testing, DenseNet121 consumed 175 ms, while VGG16 consumed 150 ms, saving 17% time. The speed of VGG16 is significantly faster than DenseNet121, because VGG16 has fewer number of layers.

For a real-world application, performance is not the only benchmark. In training phase, VGG16 cost much less money on AWS, while in web, VGG16 noticeably improves the response time. Since their performance is very close. In our web applications, we finally used our best VGG16 model as the backend model.

Software Design

The final software product will include a web application, which has an image uploading page, prediction display page, and a prediction images list page.

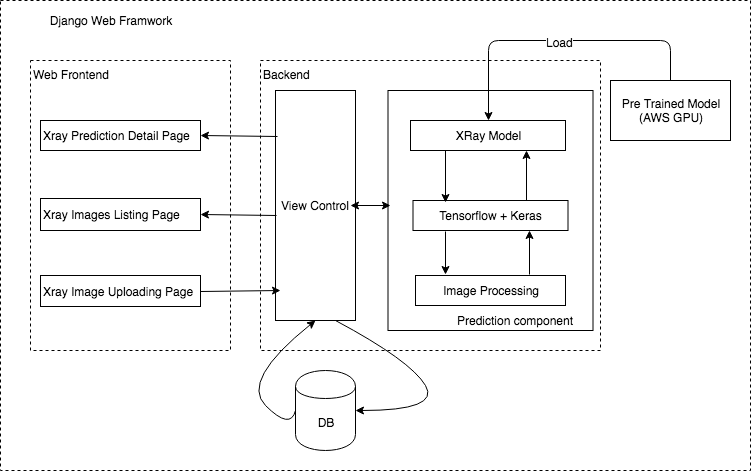


Figure 1. System Component Overview

The image uploading page allows the user to upload an image of the X-ray and then the image will be processed in the backend using the pre-trained VGG16 model. The prediction/classification result will be saved into the database and also will be displayed in the prediction display page. We also provide a prediction listing page which will present the 10 latest prediction X-Ray images together with the prediction result so the user can compare the different X-Ray images with the result. The overview of the system is shown in Figure 1.

In the web backend, we have the prediction component which will process the X-Ray images and feed the images into the pre-trained model. The TensorFlow and Keras libraries will be used as a core part of prediction component. The library provides the necessary APIs to process the images

into tensors and run pre-trained models in reference mode. The final web application is hosted in AWS, we provided the assess link for user to test our service.

Future Work

The VGG16 and DenseNet121 models have provided high performance on classifying the images into different types of diseases. We would like to use other patient information in combination of their X-ray images to further improve model performance. For example, their demographic information, such as age and gender, can be used to determine the prior probabilities of the diseases, which will lead to a better classification result.

Currently, our models are not designed to identify the problematic areas explicitly. The illustration of problematic areas could be helpful for both patients and doctors to know deeper with the chest x-ray images and better design their treatment plans if necessary. We propose to use Mask-RCNN for identifying the problematic areas. Mask-RCNN is a detection deep learning algorithm, which has been proven to have high accuracy in instance-level detection [11]. The Chest X-ray dataset already contains the location information for the disease on each image. It can be directly used to train Mask-RCNN model.

In the future, we would like to add more functionalities to our web applications. For example, adding patient profile management component to allow the user to search and manage the patient X-ray images. Additionally, we would like to allow the user to manually add actual diagnosed disease and problematic areas to the X-ray images. These results can also be saved to the database and used to train and improve the model later.

Conclusion

In this project, we built a Chest X-ray disease diagnosis web application by training deep learning models on the ChestX-ray14 dataset. To select the best backend algorithms for diagnosis, we first explored different deep learning architectures, including VGG16, VGG19, ResNet50 and DenseNet121, in the early stage. It showed that, VGG16 and DenseNet121 performed significantly better than the other two. Then we finely tuned the implementation hyper parameters on VGG16 and DenseNet121. These two models have similar performances on the test dataset. Since VGG16 is faster than Densenet121 in both training and testing, we finally selected our best VGG 16 model as the backend model.

Our application utilizes deep learning to give health care system users an easy-to-use tool in disease detection. We hope this application will help professionals in this field to improve efficiency.

References

1. Mollura, Daniel J., Ezana M. Azene, Anna Starikovsky, Aduke Thelwell, Sarah Iosifescu, Cary Kimble, Ann Polin 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, no. 7 (2010): 495-500.
2. Rajpurkar, Pranav, Jeremy Irvin, Kaylie Zhu, Brandon Yang, Hershel Mehta, Tony Duan, Daisy Ding et al. "Chexnet: Radiologist-level pneumonia detection on chest x-rays with deep learning." *arXiv preprint arXiv:1711.05225* (2017).
3. Wang, Xiaosong, Yifan Peng, Le Lu, Zhiyong Lu, Mohammadhadi Bagheri, and Ronald M. Summers. "Chestx-ray8: Hospital-scale chest x-ray database and benchmarks on weakly-supervised classification and localization of common thorax diseases." In *Computer Vision and Pattern Recognition (CVPR), 2017 IEEE Conference on*, pp. 3462-3471. IEEE, 2017.
4. Yao, Li, Eric Poblenz, Dmitry Dagunts, Ben Covington, Devon Bernard, and Kevin Lyman. "Learning to diagnose from scratch by exploiting dependencies among labels." *arXiv preprint arXiv:1710.10501* (2017).
5. Huang, Gao, Zhuang Liu, Laurens Van Der Maaten, and Kilian Q. Weinberger. "Densely connected convolutional networks." In *CVPR*, vol. 1, no. 2, p. 3. 2017.
6. Ioffe, Sergey, and Christian Szegedy. "Batch normalization: Accelerating deep network training by reducing internal covariate shift." *arXiv preprint arXiv:1502.03167* (2015).
7. Li, Zhe, Chong Wang, Mei Han, Yuan Xue, Wei Wei, Li-Jia Li, and F. Li. "Thoracic disease identification and localization with limited supervision." *arXiv preprint arXiv:1711.06373* (2017).
8. Guendel, Sebastian, Sasa Grbic, Bogdan Georgescu, Kevin Zhou, Ludwig Ritschl, Andreas Meier, and Dorin Comaniciu. "Learning to recognize abnormalities in chest x-rays with location-aware dense networks." *arXiv preprint arXiv:1803.04565* (2018).
9. Simonyan, Karen, and Andrew Zisserman. "Very deep convolutional networks for large-scale image recognition." *arXiv preprint arXiv:1409.1556* (2014).
10. Huang, Gao, Zhuang Liu, Laurens Van Der Maaten, and Kilian Q. Weinberger. "Densely connected convolutional networks." In *CVPR*, vol. 1, no. 2, p. 3. 2017.
11. He, Kaiming, Georgia Gkioxari, Piotr Dollár, and Ross Girshick. "Mask r-cnn." In *Computer Vision (ICCV), 2017 IEEE International Conference on*, pp. 2980-2988. IEEE, 2017.