BMI260 PROJECT FINAL REPORT

Yiheng Li
yyhhli@stanford.edu

Shi Chen
shic613@stanford.edu

Motivation

This project aims to identify Pneumothorax from chest X-rays. Pneumothorax is defined as the presence of air in the pleural space,[1] a life-threatening disease that is caused by external injuries, lung disease, or no obvious reason[2]. The current golden standard for diagnosing Pneumothorax is by chest radiography. However, human interpretations of images are subjective, and the accuracy of diagnosis relies on professional skills[3]. This project proposes a deep learning method to do a binary classification between normal chest radiography cases and Pneumothorax cases.

Data Set

The dataset this project used is from "SIM-ACR Pneumothorax Segmentation in Kaggle[4]". Though the data set contains training and testing set, because the testing set has no labels provided, we will only make use of the training set. Generally speaking, the data set has 12954 of examples in the format of DICOM. Each patient has a DICOM file where there is one radiographic image for one series and there is a series for one study. Each image is labeled either as '-1' for normal chest radiography, or has run-length-encoded (RLE) masks to denote the position of the Pneumothorax lesion in the image. The data set consists of 9378 of positive examples and 3576 of negative examples. Fig. 1 is an example of image data in the data set.

[1] Choi WI. Pneumothorax. *Tuberc Respir Dis (Seoul)*. 2014;76(3):99-104. doi:10.4046/trd.2014.76.3.99.
[2] Zarogoulidis P, Kioumis I, Pitsiou G, et al. Pneumothorax: from definition to diagnosis and treatment. J Thorac Dis. 2014;6(Suppl 4):S372-S376. doi:10.3978/j.issn.2072-1439.2014.09.24
[3] Ding, W., Shen, Y., Yang, J., He, X., & Zhang, M. (2011). Diagnosis of pneumothorax by radiography and ultrasonography: a meta-analysis. Chest, 140(4), 859-866.
[4] "SIIM-ACR Pneumothorax Segmentation | Kaggle." https://www.kaggle.com/c/siim-acr-pneumothorax-segmentation. Accessed 4 May. 2020.

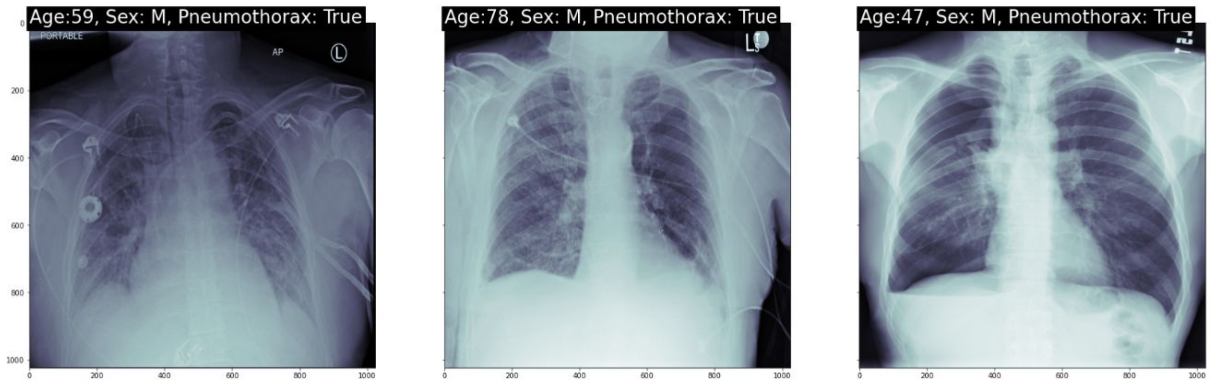
Age:59, Sex: M, Pneumothorax: True
Age:78, Sex: M, Pneumothorax: True
Age:47, Sex: M, Pneumothorax: True
PORTABLE
AP

Fig. 1 Examples of X-ray images in the Data Set.

Data Preprocessing

1. **Read in the DICOM File**. The data will be converted from DICOM format to trainable format of TensorFlow data frame, in which the image data as well as patient id data, age, gender and other information will be extracted and converted to feature space for training.
2. **Clean the Data**. samples with age > 150 or age < 14, and samples without proper labels will be excluded from the study.
3. **Downsample the Image**. Each image is stored in 1024 by 1024 pixels, which is computationally inefficient to train a neural network on. Downsampling filters including linear and non-linear filters will be used to reduce the size of the image. Images will be downsampled to 224 by 224 or 256 by 256 sizes for different training processes.
4. **Create Multi-channel Images**. By duplicating grayscale images 3 times and overlapping the matrix into 3 channels, images with shape (image_width, image_height, 3) is created for further model training process.
5. **Rescale Images**. We rescale the input Dicom images which contain pixel values from 0, 1 to be containing values of integer type and in range of [0, 255], which is standard input format for our model.
6. **Convert the Labels**. The labels of this study are assigned as 1 for positive cases of Pneumothorax and 0 for negative cases. The labels were obtained by matching the patient ids of Dicom image metadata with the rle mask notation table, where '['-1']' is assigned to 0, and other cases are assigned to 1. And patients that have no labels, i.e '' for rle masks, are excluded in the study.
7. **Data Split**. The total data which contains ??? positive case images and ??? negative case images are first splitted into training sets v.s. testing set of size 10:1. And then, in the training set, the validation set is further splitted of size 10:1 to get the final training set and validation set. The random seed is fixed and this split is valid throughout all the models mentioned further.
8. **Data restoraging.** After all the things done in the previous steps, the images are recollected and stored to new folders, where training set is stored in "data/train" folder, and positive images and negative images are in "/pos" and "/neg" folders respectively; validation set is stored in "data/val" with similar layout to training set. Test set is stored in "data/test" folder with images and labels separated and randomly sequenced.

9. **Data Augmentation**. Images are augmented to get more training examples and obtain better performance. Vertical flip is not valid in chest x-ray, so vertical flip is disabled. Other rules for data augmentation are as follows: rotation range: 20%, width shift range: 10%, heigh shift range: 10%, horizontal flip: on, shear range 10%, zoom range: 10%, brightness range: (90%~110%). To make sure the augmented images are valid, images are examined. Examples of augmented images are presented below. And before fitting into the model, images are rescaled using the preprocessing function provided by "tensorflow.keras.applications.<model>.preprocess_input" function to ensure the scale of the image matches the network.

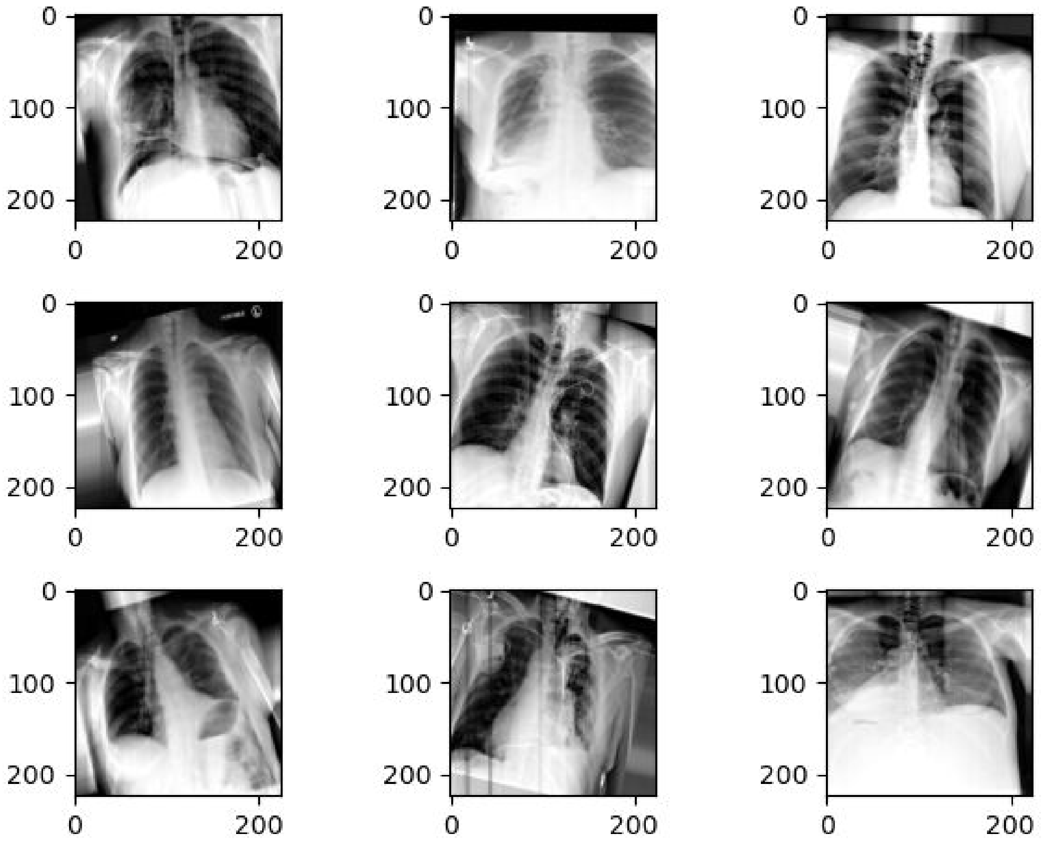

Fig. 2 Examples of Augmented Images

Model Description

Several popular image nets with pretrained weights are used in the study.

1. **VGG16**[5]: VGG16 is loaded with 'imagenet' weights. Input shapes are specified to be (224, 224, 3) and (256, 256). Output layers are excluded with 'include_top = False' parameter. New accustomed layers are added at the end of the pre-trained model to yield output. They consist of a flatten layer, a fully-connected layer with 512 nodes, a fully connected layer with 32 nodes and an

[5] Simonyan, K., & Zisserman, A. (2014). Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556.*

output layer with two nodes. The rest of FN layers are using 'relu' as activation function while the output layer uses softmax as output function. Between the flatten layer and the first FN layer, and between the first and the second FN layers, dropout layers with 0.2 as dropout rate are added to prevent overfitting issues. The model uses Adam optimizer with a learning rate of 0.001. And it used binary cross entropy as the loss function to train. During training, batch size of 32 is used, step size is calculated as the number of all training examples divided by batch size. The model is trained for 50 epochs.

2. **InceptionV3**[6]: Compared to VGG16, InceptionV3 has a more complex architecture and fewer number of parameters. Instead of stacking convolutional layers sequentially, InceptionV3 stacks Inception modules. InceptionV3 is loaded with 'imagenet' weights and the input shapes are specified to be (256, 256, 3). The top layers of pre-trained InceptionV3 are excluded and new accustomed layers are added to yield the prediction. The added layers are the same as the accustomed layers added for the VGG16 model. The model uses SGD optimizer with a learning rate of 0.001 and binary cross entropy as the loss function. The number of nodes in FC layers can be optimized by using the grid search strategy.
3. **DenseNet121**[7]: DenseNet is another popular used and widely acknowledged image net. For our application, the output layer was moved and replaced by two hidden layers with nodes equals 1024 and 128, with "relu" as activation function. The output layer is similar to previous networks. Two dropout layers are similarly added with dropout rate 0.2 to prevent overfitting issue, which did show in training densenet.
4. **EfficientNetB5**[8]: EfficientNet is a lightweight convolutional neural network architecture achieving state-of-the-art accuracy with an order of magnitude fewer parameters and FLOPS, on both ImageNet and five other commonly used transfer learning datasets. We want to explore the potential of this very new (released in 2019) architecture on our task. The whole family contains B0 - B7 nets. As the version number increases, the amount of parameters increases with the accuracy of the imagenet dataset. We choose to use version B5 as our model, to balance the computing efficiency lost with model accuracy gained. Similarly, the model is loaded with fixed weights and the output layer is removed when import. The weights inside the model structure are set to be non-trainable. Two fully-connected layers which optimal node number tested by VGG16 net are added before the final output layer which contains two is added. Weights are initialized and will be fine tuned using the training data set. The parameter of the model is much bigger compared to densenet and VGG net, so a smaller batch size of 8 is set, and the training process consists of 30 epochs.

[6] Szegedy, C., Vanhoucke, V., Ioffe, S., Shlens, J., & Wojna, Z. (2016). Rethinking the inception architecture for computer vision. In *Proceedings of the IEEE conference on computer vision and pattern recognition* (pp. 2818-2826).
[7] Huang, G., Liu, Z., Van Der Maaten, L., & Weinberger, K. Q. (2017). Densely connected convolutional networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition* (pp. 4700-4708).
[8]Tan, M., & Le, Q. V. (2019). Efficientnet: Rethinking model scaling for convolutional neural networks. *arXiv preprint arXiv:1905.11946*.

Training Settings: There are a few things that are worth mentioning during the training process. The results are shown using image shape (224, 224, 3) except for InceptionV3, which remains (256, 256, 3). The dropout layers are only added if an overfitting issue occurs. Except for the EfficientNet-based model, all other models have dropout layers added. Class weights are setted to resemble the proportion of positive cases and negative cases in our training dataset, in order to deal with the imbalanced learning problem. All the models are trained with Adam optimizer[9] with default parameters. Binary cross entropy is selected for all models as the loss function. To prevent last layers from overfitting, a regularizer with l1_l2 panelizer parameter 0.01, 0.01 is set over the weights connecting the last hidden fully-connected layers.

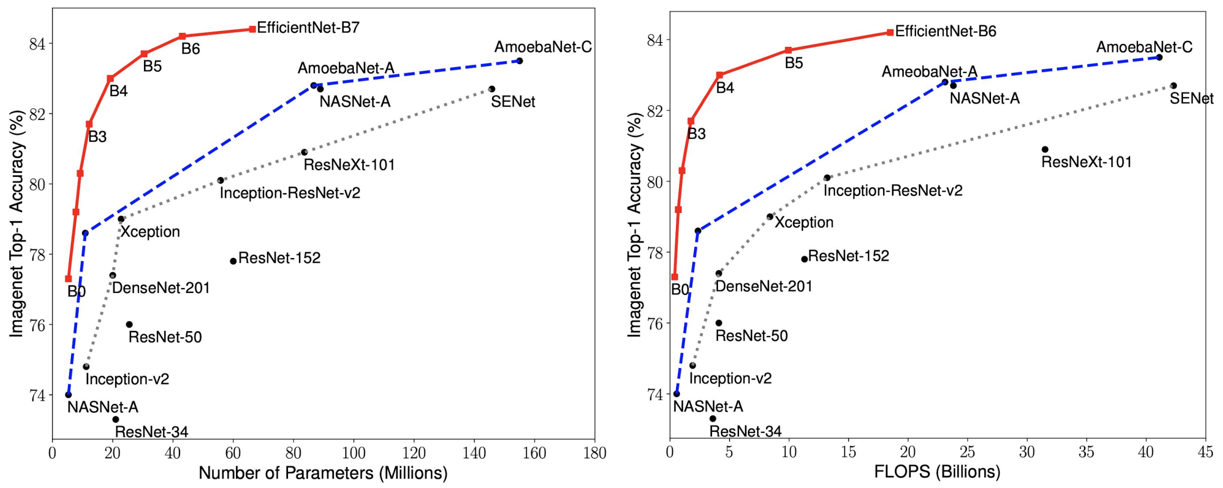

Fig 3. The Complexity and Accuracy Comparison of Popular Image Nets

Results

1. VGG16: VGG16 is an early convolutional net, which is comparatively fast to train on. On it, hyperparameters of the number of nodes in the last two hidden layers are searched. The result is shown below.

ACCURACY		Number of nodes in the 1st fc layer			
		128	256	512	1024
Number of nodes in the 2nd fc layer	16	73.85%	50%	77.16%	72.74%
	32	76.43%	77.16%	78.45%	77.34%
	64	73.11%	75.87%	77.90%	71.82%

Table 1. Hyperparameter Tuning (# of Nodes in the Last Two Hidden Layers) Results (Accuracy) of VGG16-based Model

[9] Kingma, D. P., & Ba, J. (2014). Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*.

After setting the structure for the last layers, the model is trained on the training set and tested on the testing set. The loss curve is shown as below.

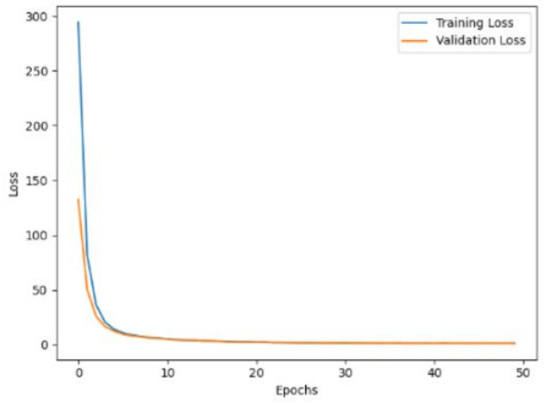

Fig 4. Loss Curve of VGG16-based Model for 50 Epochs

The performance of the model achieves accuracy: 70.06%; AUCROC:79.31%. The confusion matrix of the model on the testing set is presented in the table below.

		Truth condition	
		Pneumothorax	Healthy
Test result	Positive	623	41
	Negative	315	230

Table 2. Confusion Matrix on Testing Set of VGG16-based Model

From the confusion matrix, the specificity is 66.41% and the sensitivity is 84.87%..

2. InceptionV3:The loss curve is shown below.

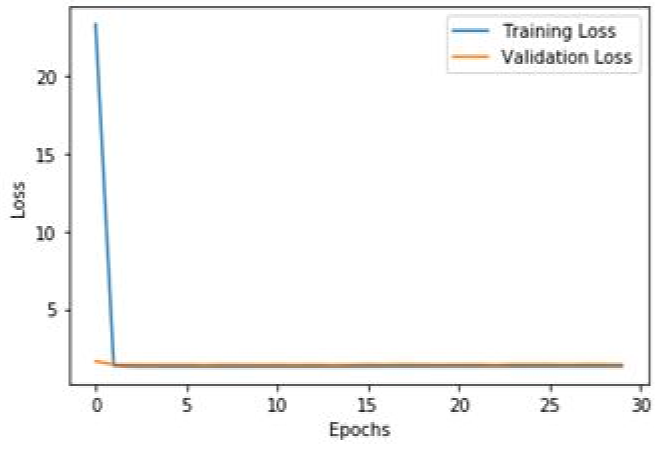

Fig 5. Loss Curve of InceptionV3-based Model for 30 Epochs

The performance of the model achieves accuracy: 49.91%; AUCROC: 49.96%.

		Truth condition	
		Pneumothorax	Healthy
Test result	Positive	0	0
	Negative	938	271

Table 3. Confusion Matrix on Testing Set of InceptionV3-based Model

3. DenseNet121: The loss curve is shown below.

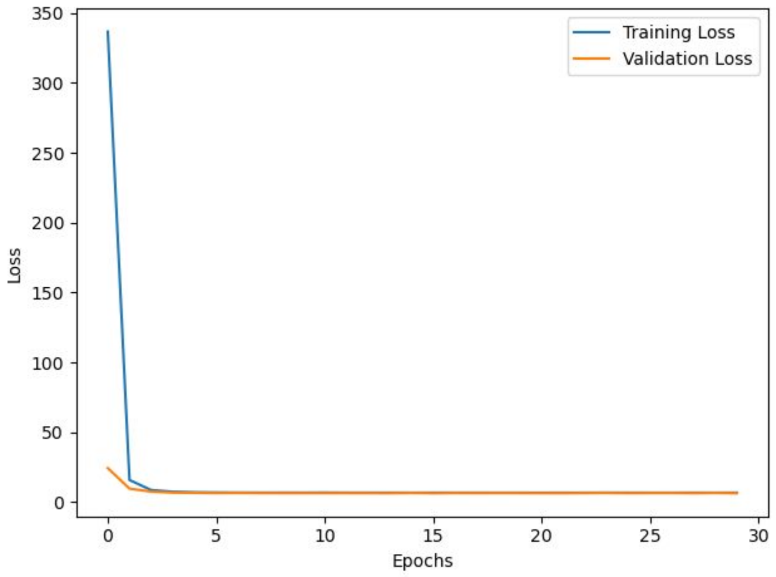

Fig 5. Loss Curve of DenseNet121-based Model for 30 Epochs

The model achieves the performance of accuracy: 80.07%; AUCROC: 68.13%

		Truth condition	
		Pneumothorax	Healthy
Test result	Positive	842	145
	Negative	96	126

Table 3. Confusion Matrix on Testing Set of DenseNet121-based Model

From the confusion matrix, the specificity is 89.77% and the sensitivity is 46.49%.

4. EfficientNetB5:

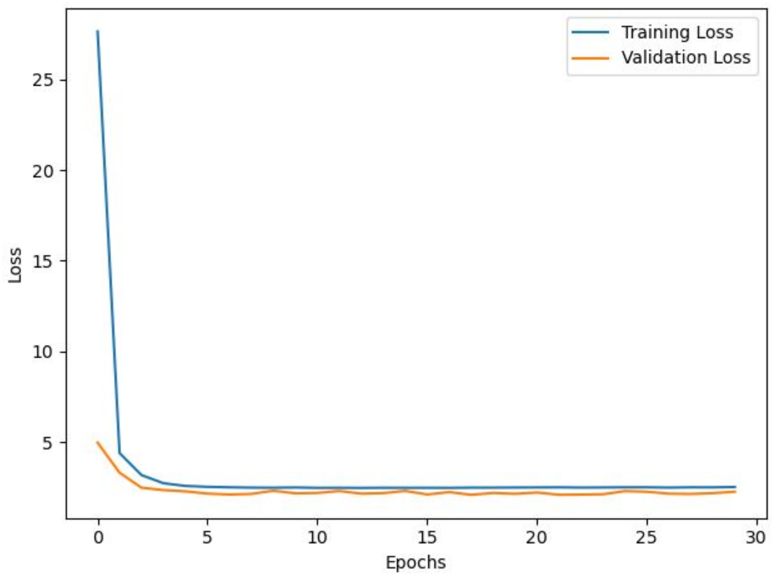

Fig 6. Loss Curve of EfficientNetB5-based Model for 30 Epochs

The model achieves the performance of accuracy: 71.13%; AUCROC: 73.39%

		Truth condition	
		Pneumothorax	Healthy
Test result	Positive	650	61
	Negative	288	210

Table 4. Confusion Matrix on Testing Set of DenseNet121-based Model

From the confusion matrix, the specificity is 69.30% and the sensitivity is 77.49%.

Discussion

In this study, several 2D deep learning models were developed from popular pre-trained convolutional models and preprocessed chest radiography were used as the inputs for models to predict Pneumothorax. Four models were developed and tested on the test set.

The first model based on VGG16 achieved around 70% accuracy on the test set. From the confusion matrix given by the first model, it can be calculated that the specificity is lower than the sensitivity. It indicated that the model cannot predict Pneumothorax well given the patient actually has Pneumothorax. Given that Pneumothorax may progress to a life-threatening disease, it was problematic that the specificity is low because the false negative prediction may prevent patients getting immediate help and even lead to an irreversible condition.

The second model based on InceptionV3 achieved around 50% accuracy on the test set. It implies that the model did not learn features useful to make a prediction. From the confusion matrix given by the second model, it was clear that the model predicted negative for all test data and it confirmed that the model cannot make useful predictions. It was hard to explain why the InceptionV3-based model cannot generalize to the test data and one possible answer was that this model was not suitable for this application. More effort could be used to investigate the low performance of the model and it could give some useful insights on how to choose the base model.

The third model based on DenseNet121 achieved around 80% accuracy. From its confusion matrix, the specificity given by this model is higher than the sensitivity. And the specificity of this model was also higher than the specificity of the VGG16-based model. In this application, the high specificity could lead to more patients with Pneumothorax to get immediate help and prevent the progress of the disease. The high sensitivity raised some concerns on the economic aspect because it could increase the cost of the patient or health institutions by spending unnecessary time and efforts on healthy people who are predicted to have Pneumothorax by the model.

The fourth model based on EfficientNetB5 achieved around 71% accuracy, which was close to the accuracy by the VGG16-based model. Compared to the VGG16-based model, this model gave slightly higher specificity and lower sensitivity.

Among all the performances of four models, the model based on DenseNet121 performed much better in terms of accuracy and specificity.

Conclusion

This study showed that it is feasible to develop a 2D deep learning model on chest radiography to predict Pneumothorax. The model can be developed using transfer learning by using existing 2D convolutional models with some hyperparameter optimization. If more time permitted, the future works of this study could be in the following directions:

1. Try out more pre-trained models. Though we have tried the majority of most popular image models out there, there are still some of them we did not use. Each model has the opportunity to outperform previous models. Trying out more models is definitely worthwhile if we want to optimize the performance.
2. More flexible hyperparameter tuning. The hyperparameters that were tuned and considered in this study include nodes of several last hidden layers, dropout layers, learning rate, input shape, etc. There are still many other hyperparameters to tune besides those, including those for fastening training process, e.g. Parameters in Adam optimizer, batch size, batch normalization, etc and those for general performance improvement, e.g. activation functions, number of fn layers before output layer, etc.
3. Train for more epochs. Though the training process slowed significantly after 30 or 50 epochs of training. There should be space for a very deep neural network to make progress in hundreds of epochs. More patience in training, or using more advanced machines for training bigger epochs would optimize the performance of the model gradually.

Code

Codes for this project is provided here:
https://github.com/terryli710/SIIM-ACR-Pneumothorax-Classification

Contribution

Yiheng Li built the pipeline for image data preprocessing, label constructing and data storage, and he implemented data augmentation with the TensorFlow package . He also trained and tested VGG16-based model, DenseNet161-based model as well as EfficientNet-based model. **Shi Chen** constructed base codes for transfer learning of this task using TensorFlow, and he implemented the hyperparameter search of the number of nodes in the last two hidden layers based on VGG16. He also trained and tested VGG16-based model and InceptionV3-based model. We want to make our acknowledgements to our TA **Mars Huang**, who provided the general picture of this project and patiently gave us very helpful advices and guidance during the project.