X-Ray Image Classification - Final

Akshay Raul, Brent Younce, Vishnu Nadimpalli, Bradley Beach
North Carolina State University
araul, bjyounce, lnadimp, bcbeach2 @ncsu.edu

Abstract

In this study of X-Ray image classification, we compare, analyze and contrast 3 machine learning techniques to classify Chest X-Ray images according to disorder categories. We used the ChestX-ray14[1] X-Ray image dataset, along with the corresponding research from the NIH to understand the dataset and its labels. We compared and analyzed Random Forests, Shallow CNN and Deep CNN classifiers to understand the advantages and disadvantages of each architecture on this problem. Through this study we aim to understand the effectiveness of each of the models, reporting the accuracy, f1- scores and choosing which model is best suited for X-Ray image classification, which has many interesting considerations such as explainability, the differences in importance of recall vs precision, and the complexity in image interpretation. The experimental code is hosted on GitHub. [2]

1 Background and introduction

1.1 Problem Introduction

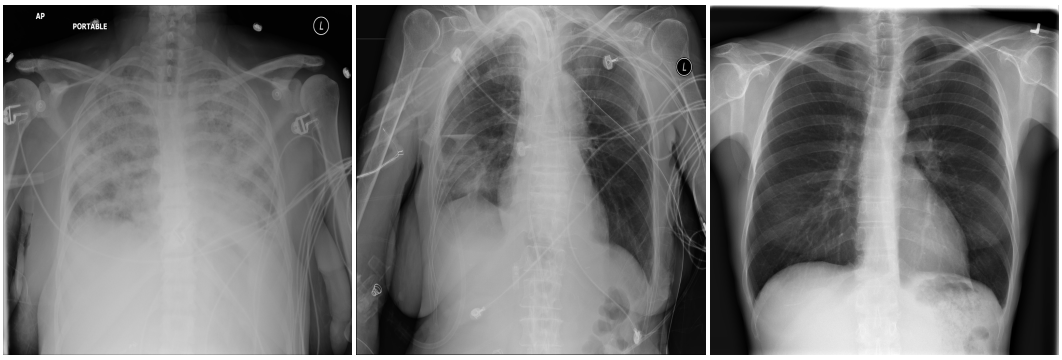
There is currently a large and growing shortage of medical professionals (Dall et al. [2018]). Because of this, there is a constantly increasing backlog of medical test results that have yet to be examined. While some tests are straight-forward and can be handled by nurses, such as comparing a blood pressure measurement to the known healthy range, other tests take specialists to comprehend. If someone is not trained to analyze x-ray photos, for instance, they will not be able to tell a healthy lung from a diseased lung, let alone discriminate between diseases. The labor shortage is compounded by these difficult-to-interpret tests and the massive numbers of images produced by x-ray and other scanning machines, leading to long turnarounds for people suffering from time-critical diseases.

It takes years to train a radiologist; the backlog will continue to grow for quite some time if society waits for the next generation of doctors to come along, and if the demand for radiologists keeps growing at current rates, the next generation will fall even further behind. Thanks to state of the art of computer vision, however, there may be another solution. Image classifiers based on machine learning can be developed and trained in a fraction of the time it takes to teach a person to read x-rays. Once made, they are easy to deploy on a large scale. If an x-ray classifier was developed with performance equivalent to a radiologist, it could make great strides towards faster diagnoses for patients. Classifiers could be used as a first line of x-ray analysis, telling the radiologist which images require examination and which images don't appear to show anything of value.

For this project, we chose the Chest-X-ray14 Hospital-scale Chest X-ray Database (Wang et al. [2017]), which contains 112,120 chest x-rays in 1024x1024 resolution. 14 different diseases are present among the x-rays. When combined with a with healthy patient group, this results in 15 possible classes. Although the images are labeled with their appropriate diagnoses, it is important to note that these are not entirely accurate. In addition to the standard human error present in x-ray evaluation, the diagnoses for the pictures were scraped from medical reports via Natural Language Processing. Since this data comes from real patients, it is not always simple; diseases are not mutually

[1] https://arxiv.org/abs/1705.02315
[2] https://github.ncsu.edu/araul/CSC522_Project

exclusive, and one x-ray may be classified as containing several of them. One additional quirk of this dataset, from a machine learning perspective, is that not all x-rays contain enough information to reach a decision. A doctor has the option of ordering additional x-rays or a battery of tests to help diagnose a patient, but the classifier will only ever have the single provided x-ray.



Sample X-Ray Images from our dataset

For this project, our goal is to compare three types of classifiers: Random Forest, Shallow CNNs, and Deep Neural Networks in classifying an X-Ray as either representative of a condition or of a healthy patient. As a stretch goal, we would like to investigate the possibility of a classifier displaying which disorder an X-Ray corresponds to, as well. One final important consideration for any classifier is that, in medical contexts, false negatives (determining that an X-Ray does not correspond to a disorder when, in reality, it does) results in patients not being treated, while false positives (determining an X-Ray does correspond to a disorder when it does not) tend to result only in more tests. Therefore, any classifier should highly prioritize improving recall over precision.

1.2 Background

To determine the best approaches to this problem, a literature search was performed, with a focus on classifier training methods. Through this, three machine learning systems were identified as viable targets for exploration. Bosch et al. [2007] showed that, for data sets with multiple categories, improvements could be made through use of random forests. In a random forest classifier, multiple decision trees are generated, with some element of randomness influencing each tree's structure. To classify a data point, each of the trees is given a vote on the data; as the size of the forest increases, the effectiveness of the classifier improves, and the requirement of expert input is minimized. Methods based on decision trees, like this one, are useful in a medical context because the decision of the classifier can be justified to the doctor by showing them the decision trees, which are already in an easy-to-follow format.

The explanation of convolutional neural networks in Lecun et al. [1998] described similar benefits. CNNs, which are based on the structure of neurons in a brain, allow for the features to be extracted from the data without being hand-specified by an expert. An expert's knowledge was traditionally the limiting factor in constructing machine learning systems. Especially when dealing with natural data with uncountable minuscule differences, it becomes very difficult for an expert to put their years of experience into words; while they can list the steps they take in making a decision, some parts of an expert's job are second-nature to them, and nuances can easily be overlooked. CNNs learn the patterns themselves, leading to better patterns faster. CNNs tend to provide better results than simpler methods, but are not so complicated that they are incomprehensible to a layman.

Finally, He et al. [2005] detailed the methods used to win multiple image classification challenges. Previously, the vanishing gradient problem limited the number of layers that can be included in a neural network. This difficulty was removed through the use of normalization, both during initialization and in normalization layers interspersed with the rest of the network. Next, a degradation problem, separate from the risk of overfitting, appeared. He's group solved this problem through the use of deep residual learning, shortcut connections sending identity mappings forward. With the degradation problem conquered, their deep residual networks achieved excellent error rates. Deep neural nets clearly show potential for generating strong results in x-ray classification. Our approach includes two recent variant architectures. The DenseNet (Iandola et al. [2014]) architecture,

which lowers computational needs by sharing the workload among overlapping regions, is useful in situations where the object regions are particularly dense. This may be useful for medical situations, where the object regions of interest may be everywhere, nowhere, or somewhere in between. Wide Residual Networks (Zagoruyko and Komodakis [2017]) are another good candidate for our purposes. Wide ResNets are a literal twist on traditional deep neural networks, built to be wide and shallow as opposed to thin and deep. Wide ResNets have reported state-of-the-art accuracy in a fraction of the runtime needed by competitive deep networks.

2 Method

In order to accomplish the above goals, as well as explore a wide range of potential avenues for solving the aforementioned problems, we have decided to take a multi-pronged approach. That is, we will train three different types of machine learning systems, and compare and analyze their performance.

2.1 Random Forest

The first type of machine learning system we will train to automatically label x-ray images with their diagnoses is a random forest of decision trees. This approach has the advantage of being simple to visualize and explain to patients, doctors, and other system stakeholders. Especially in the medical field, where stakes for accurate diagnoses are extremely important, explainablity is a major advantage. Specifically, we will use the RandomForestClassifier from Scikit Learn and train the images with multiple labels. Random Forests work particularly well with defined labels. We fit the RandomForestClassifier with training data of images. The data consists of pixels of each image(1024*1024 pixel) in a flatten array passed on to the classifier. Then the classifier randomly picks m out of K features and calculate the best split point. A comprehensive study will be conducted into the architecture of RandomForestClassifier and we will tune the hyper parameters of the classifier to get the best output. We also need to consider that RandomForest classifiers solely take into account pixel value at a position, hence it does not take into account the view positions, how many bits the image has and if it is multi-labeled. As our images may have many diagnoses, and correspond to both anterior and posterior views, these are important considerations. Additionally, we must ensure that underfitting or overfitting are monitored. After the model is created and fit with the training data, we then use the model to predict results for a separate testing dataset.

2.2 Custom Shallow CNN

In addition, we will also train a custom, shallow CNN to label x-ray images with their appropriate diagnoses. This setup provides a solid middle-ground between more explainable decision trees and less explainable deep neural nets. Shallow CNNs are fairly simple to visualize, fast to train, and well understood, properties which lend themselves well to this problem. Specifically, we will train a Tensorflow-based convolutional neural network through the Keras library based on MobileNet Architecture (Howard et al. [2017]). We intend to design the neural net to have a limited amount of layers, to aid in explainability and training speed.

A comprehensive study was conducted to find out the best additions to the base MobileNet architecture. A normal 2D convolutional layer(grey scale) contains an optimally sized kernel and this kernel would help in identifying the sub-features of the images and their spatial orientation in the images. In the case of MobileNet architecture, this normal convolution is replaced by depthwise convolution followed by pointwise convolution which is called depthwise separable convolution. This results in a significant reduction in the number of parameters and thereby reduces the total number of floating point multiplication operations. This architecture also contains a Leaky ReLU activation function layer which would help the network learn nonlinear decision boundaries. We implement Leaky ReLU as an attempt to fix the dying ReLU problem if it were to occur. Along with these two layers, we add a max-pooling layer which helps to reduce the variance and computational complexity and extract low level features from the neighbourhoods. In the case of deeper architectures, all these layers are repeated in certain patterns to obtain and learn the features of images at different complexities. The last layer is generally a fully connected layer with a suitable activation function and has the same number of units as the number of expected output classes. After building the network, we will finally train the model with a suitable number of epochs and validate the model with different metrics.

2.3 Deep Neural Nets

Finally, to take advantage of highly successful deep neural net designs, we will train several deep neural network models, based on architectures which have had great success in image classification, specifically on the ImageNet dataset. This allows us to take advantage of highly successful neural net architectures as well as compare our other machine learning methods to the latest in the quickly evolving neural net architecture space. Specifically, we would like to train neural networks based on the ResNet (He et al. [2005]), Inception, Wide ResNet and DenseNet architectures, using a wide variety of hyperparameters, dataset samples, and class weights.

3 Experimental set-up

3.1 Decision Tree

The Scikit Learn Random Forest Classifier was utilized to form multiple decision trees, as we are dealing with multi-label data. From there, the following tuning parameters were used: n_estimators=10, max_depth=4, bootstrap=True, n_jobs=2, criterion=Entropy.

We utilized the following libraries for this classifier:

- Scikit Learn- for model training and setting up Random Forest Classifier. Random Forest Classifier in Scikit provides many parameters which them can be coupled woth GridSearchCV and other estimators
- Jupyter Notebook- For the simplicity of development, ability to write clean code and the ability to produce visually well formatted explanations
- Pandas- Popular data science library which provides high performance tools and data structures

3.2 Custom Shallow CNN

Training a CNN with such a huge data set would be extremely time consuming locally. Hence, in order to support the training of our neural networks in reasonable periods of time, we have utilised the large GPU power available from Google's Cloud Compute platform. Here, We have created an instance with a NVIDIA tesla K80 GPU and set up CUDA NN libraries from Nvidia. Next, we installed Tensor-gpu libraries and all other supporting packages such as numpy, pandas, scipy, keras etc. to work with the dataset. Also, Jupyter notebooks were also utilized for development, as they provide a great interface to interact with instance and allows for the development and documentation of the code in one place.

Data manipulation also plays a vital role in improving the quality of our predictions. Firstly, we create a pandas dataframe which includes the image's filename, label and path. For this model, the other attributes present in the dataset are not applicable, and so are ignored. This dataset, as mentioned, contains a large number of images, each with potentially multiple labels. One way to deal with this would be to take the labels and and create a one hot encoding array for all significant outputs. To simplify the problem, we are omitting classes with labels which make a very small fraction of the dataset. From this, for this model, we have reduced the dataset to objects with primarily five labels i.e five diseases. One of the advantages of using keras library is the tools it provides for this data wrangling process. We can use packages like ImageDataGenerator to produce additional variability in the data by performing geometric operations like flipping horizontally, vertically, introducing shear, rotation etc. Finally, we split the data into training, validation and test sets with desirable proportions and input the images as flattened array of pixel values to the neural net.

3.3 Deep Neural Nets

In order to support the training of these networks, which often require large quantities of GPU power, these models will be trained on a Google Cloud Compute instance with a mixture of NVIDIA Tesla K80 and P100 GPUs (dependant on the computational needs of the particular architecture), which provide extremely high computational capabilities as compared to local CPU training. In terms of software libraries, the following will be utilized:

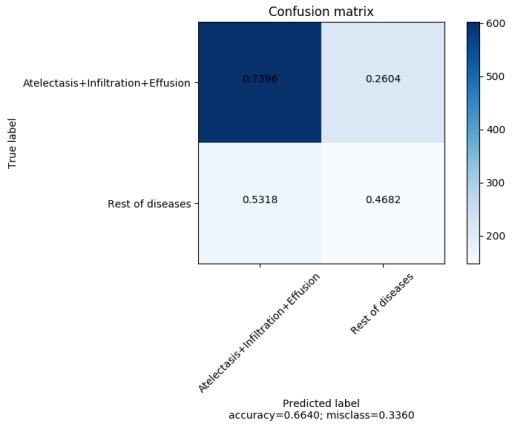
- PyTorch - for model training. Selected due to its native support for the architectures we intend to explore, and its clean API
- FastAI - for dataset augmentation and learning rate finding
- Jupyter Notebook - To provide a clean development interface and environment to produce well-formatted and explainable code

4 Results

4.1 Random Forest

We began by initially building a random forest of trees based on a random sub-sample of about 10000 images. With this small sample, the accuracy of model tends to be lower than what we need to properly assist radiologists. For 10000 images, this model produced an accuracy of about 42%.

Accuracy, however, is not a highly valid metric in this case, due to the highly unbalanced nature of the dataset (with 80-95% No Finding results, depending on the random sample). The accuracy should also increase when we implement cross validation for the given dataset. Since Random Forest classifiers are not great at recognizing large numbers of sub-features in complex images, they might not produce accurate results, but they do greatly help with correlating images. According to Wang et al. [2017], diseases are co-related, and not necessarily independently occurring in a particularly area in the chest. The study shows Atelectasis, Effusion, Infiltration and Mass are highly correlated. Random Forests greatly exhibit this correlation as the predicted labels are mapped to the disease which is highly correlated. If such diseases whose acting area on the chest has similar covered area were to be clubbed together and the images are classified to fall in "No Findings" and this set of images, then the accuracy and f1-score increases substantially. The accuracy score is about 65% and weighted f-1 score of about 67%



(a) Figure 1

4.2 Custom Shallow CNN

Taking the complexity and disorderliness of the data-set into consideration, much preprocessing of the data was required before the model could be trained. The procedure discussed in the experimental setup section was followed. Next, the image data input was passed to the standard MobileNet architecture in the keras library. The initial weights used in the architecture were pretrained weights used originally for the ImageNet dataset. Next, it was observed that the addition of dropout layers and a final dense layer of weights greatly improved the accuracy for this multi-label classification problem. Finally, a sample was taken of the dataset to balance the classes, as the dataset is, by default, heavily unabalanced, with most of the images corresponding to "No Finding". After making the necessary changes to the base model and sampling the dataset, after training, it was observed that the model was able to obtain an accuracy of up to 72%. Given that the model is trained to classify 5 different disorders, this could serve as a solid middle ground between the explainability of Random Forest classifiers and the accuracy of Deep Neural Networks.

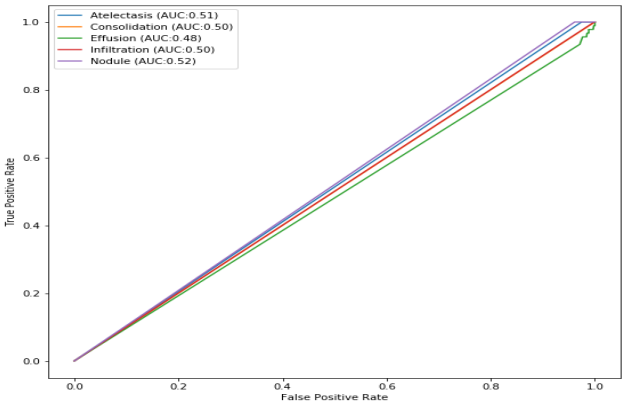


FIG 1. ROC curves for MobileNet Architecture

4.3 Deep Neural Nets

To begin the analysis of the effectiveness of deep neural nets on this problem, we began by analyzing the performance of several architectures on a slightly reduced subset of the problem space. That is, the identification of x-rays which correspond to a diagnosis of Atelectasis, the most common diagnosis in our dataset. Additionally, the officially provided sample dataset (consisting of 5% of the overall dataset, randomly selected) was used for training and validation to reduce training time. After model training was performed on this subset, highly performant models were re-trained on the full dataset

One other important consideration was that of determining performance metrics to use. Although accuracy is one metric that could be used (accuracy of determining No Finding or Atelectasis), due to the highly unbalanced nature of our dataset, that metric was deemed insufficient. That is, 93% of the samples in our dataset are labeled No Finding. Based on this fact, a trivial classifier which simply always returns No Finding would technically have a 93% accuracy value, but it would be useless in practice. In order to demonstrate the value of the models we train, we therefore provide full confusion matrices for highly performant models, and strongly focus on maximizing recall, as mentioned earlier. Additionally, to minimize false negatives which could result in patients not being treated, we trained models with a variety of class weights, ranging from 1:1 to 75:1. These refer to the preference of the model for reducing false negatives over reducing false positives

With these considerations in mind, many models were trained and evaluated. For each trained model, a similar process was used for training. First, an architecture was chosen (out of ResNet18, ResNet34, Wide ResNet, and DenseNet121). Next, a learning rate was determined following the procedure described by Smith [2017]. Next, a fully-connected layer of weights was added, and training was performed solely on this layer of weights. The default weight values from the model provided by PyTorch (typically trained on ImageNet) were used in this step. Finally, the model was "unfrozen" (allowing for the training of the full model), and training was again performed.

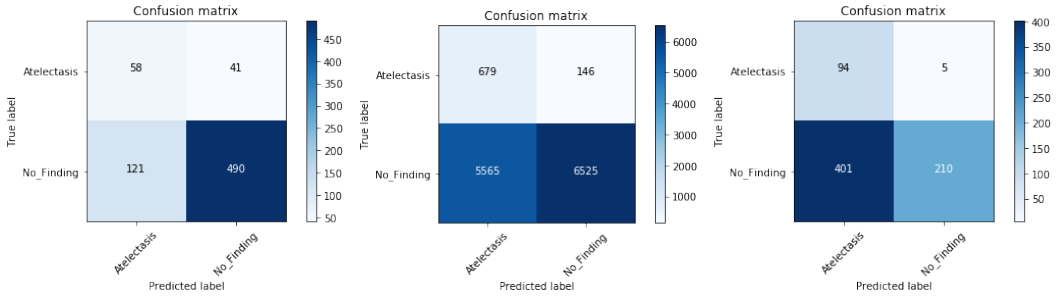
Below is a small subset of our training results, depicting the weights used (of the form Atelectasis : No Finding), learning rate, number of epochs, and other values for models trained using a variety of architectures. These demonstrate the effects of adjusting the class weights and the model architecture. In general, these models achieved fairly high levels of accuracy (in the 0.7-0.9 range), which, given the many quality issues of our dataset, is nearly as high as can be achieved with this data alone. Also, these models demonstrate great flexibility in balancing recall vs precision, based on the needs of a particular medical context.

4.4 Comparison

These three model categories clearly produced interesting and varied results for this problem. Firstly, our Random Forest model, as mentioned earlier, achieved only around 42% accuracy overall. However, the model was able to place 78% of No Finding images correctly. Although this level of accuracy would be unacceptable for directly assisting radiologists (our primary goal), the model's very high

Weight	Model	Frozen/Unfrozen	Epochs	Accuracy	Confusion Matrix	Dataset
1:1	resnet18	frozen	20	0.847		sample
1:1	resnet18	unfrozen	10	0.86619		sample
10:1	resnet18	frozen	100	0.8		sample
10:1	resnet18	unfrozen	115	0.7859	Figure 1	sample
50:1	densenet121	frozen	128	0.55	Figure 2	full
75:1	densenet121	unfrozen	128	0.43	Figure 3	sample

Table 1: Results of Deep CNN models; Learning Rate=0.01



Figures 1-3

performance in terms of execution time combined with its reasonable accuracy could yield interesting applications for saving time for radiologists. For example, such a model could be used for setting default classes in X-Ray software. We expect, with more fine-tuning, the random forest model could be improved in terms of accuracy slightly. However, due to the lack of flexibility of this model type, its main advantages seem to come in the form of its fast execution time rather than pure accuracy.

Similarly, our shallow CNN model provided low overall accuracy, comparable in many cases to random guessing. Therefore, it would also not be capable of directly assisting radiologists. We believe this is also due to the model not being complex enough to be able to take into account the full amount of features necessary to properly classify medical XRays. This model does, however, boast very fast training times (< 45 minutes), and high flexibility due to the ability to easily modify class weights.

Finally, our Deep NN architectures did actually provide fairly high levels of accuracy, with the help of the DenseNet architecture and heavy weighting towards reducing false negatives. These models, however, suffer from a fairly high number of false positives. Despite this, their high levels of recall mean they would be ideal candidates for answering our primary question. That is, a deep Neural Network, specifically one which utilizes a DenseNet or Wide ResNet architecture, could be used to directly aid radiologists in more quickly diagnosing patients. As these models have high levels of recall, they could quickly eliminate many true negative scans, allowing the radiologists to focus solely on a much smaller subset of scans which may actually demonstrate the presence of a disorder. Furthermore, we believe the high number of false positives could be heavily reduced by training using a much cleaner dataset. That is, since our dataset contained a large number of images for which there was much uncertainy in the labels, and the labels themselves were applied using imperfect (according to those who assembled the dataset themselves) NLP techniques, the level of performance we were able to achieve was capped at a level we believe to be very close to what we achieved. With a cleaner dataset, a much higher level of precision may also be achievable, while maintaining the high level of recall of our models.

5 Conclusion

To conclude, this project attempted to compare several categories of image classifiers in the field of medical image classification. Specifically,we aimed to compare Random Forest, Shallow CNN, and Deep Neural Network classifiers in classifying chest XRays as being evidence of the presence of disorders or of a healthy patient. We were able to train and compare models of each of the three categories, ending with a comparison which demonstrated the various strengths of each methodology.

Through the project, we learned the advantages of using remote development environments (such as Jupyter Notebooks on Google Cloud or AWS) in terms of cost and performance, the complexity of the features which must be modeled for this type of problem, and the challenges in working with high levels of labeling inaccuracy in a dataset. We also ended with several ideas for future work. Firstly, training a DenseNet model for much longer periods of time (on the order of a handful of days to a week) with a lower learning rate (potentially .001 on the later layers, with a gradient down to .0003 on the earlier layers) may produce slightly better results. Secondly, training a similar architecture on a much cleaner dataset may boost precision substantially. However, as a dataset that is both extremely large and extremely clean is most likely not available, we propose training a DenseNet model identical to one we produced, with the exception of a lower weight for reducing false negatives (as high weights of this type may be unnecessary after the next step), followed by fine-tuning the model on any available small but high quality datasets. This fine tuning step would allow for a researcher to take advantage of the large size of our dataset as well as simultaneously taking advantage of the high level of cleanliness of the smaller datsets. Finally, it may be beneficial for future researchers to attempt to integrate transfer learning techniques to partially mitigate the dataset quality issues.

References

A. Bosch, A. Zisserman, and X. Munoz. Image classification using random forests and ferns. In *IEEE International Conference on Computer Vision*, 2007. URL `http://www.cs.huji.ac.il/~daphna/course/CoursePapers/bosch07a.pdf`.

T. Dall, T. West, R. Chakrabarti, R. Reynolds, and W. Iacobucci. 2018 update: The complexities of physician supply and demand: Projects from 2016 to 2030: Final report. Technical report, IHS Markit Ltd., March 2018. URL `https://aamc-black.global.ssl.fastly.net/production/media/filer_public/85/d7/85d7b689-f417-4ef0-97fb-ecc129836829/aamc_2018_workforce_projections_update_april_11_2018.pdf`.

K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. Technical report, Microsoft Research, 2005. URL `https://arxiv.org/abs/1512.03385`.

A. G. Howard, M. Zhu, B. Chen, D. Kalenichenko, W. Wang, T. Weyand, M. Andreetto, and H. Adam. Mobilenets: Efficient convolutional neural networks for mobile vision applications. *CoRR*, abs/1704.04861, 2017. URL `http://arxiv.org/abs/1704.04861`.

F. Iandola, M. Moskewicz, S. Karayev, R. Girshick, T. Darrell, and K. Keutzer. Densenet: Implementing efficient convnet descriptor pyramids. Technical report, University of California, Berkeley, April 2014. URL `https://arxiv.org/abs/1404.1869`.

Y. Lecun, L. Bottou, Y. Bengio, and P. Haffner. Gradient-based learning applied to document recognition. In *Proceedings of the IEEE*, November 1998. URL `http://yann.lecun.com/exdb/publis/pdf/lecun-01a.pdf`.

L. N. Smith. Cyclical learning rates for training neural network. In *IEEE Winter Conference on Applications of Computer Vision*, 2017. URL `https://arxiv.org/abs/1506.01186`.

X. Wang, Y. Peng, L. Lu, Z. Lu, M. Bagheri, and R. M. Summers. Chestx-ray8: Hospital-scale chest x-ray database and benchmarks on weakly-supervised classification and localization of common thorax diseases. December 2017. URL `https://arxiv.org/abs/1705.02315v5`.

S. Zagoruyko and N. Komodakis. Wide residual networks. Technical report, Université Paris-Est, École des Ponts ParisTech, June 2017. URL `https://arxiv.org/abs/1605.07146`.

6 GitHub Link

`https://github.ncsu.edu/araul/CSC522_Project`