NANYANG TECHNOLOGICAL UNIVERSITY SINGAPORE

Masters in science (Artificial Intelligence) (MSAI)

AI-6102

Assignment 3

Name of Student:
Teo Lim Fong

S210024@e.ntu.edu.sg

Matriculation No: G2101964G

Contents

Introduction

Chest X-ray imaging is an important diagnosis tool for preventing someone life in risk. However, due to the shortage of radiologists, we must delay the treatment for some of the patients which is not an ideal solution. The patient may be in critical condition and if an event of huge outbreak of diseases such as the recent Covid-19. We required huge number of radiologists to examine if the patients are affected with Covid-19. Therefore, in this assignment I will be incorporating an AI Technologies that uses both machine learning and computer vision to predict the possible diagnosis labels in the testing chest x-ray to solve the shortage of radiologist's issue and later addressing a few of the AI Responsible issues.

Application Description

Let's assume you are the chief of Medical Officer in ABC hospital, and you want to solve the shortage of radiologist's issue by using AI to predict the diagnosis label of the chest x-ray. And since we have more than enough datasets, we can ask a data scientist to help us train a model ***shown in Figure 1*** to predict the diagnosis labels in the testing chest x-ray image ***shown in Figure 2***. This way, we can efficiently address those patients that diagnose with lungs issue and preventing people from serval life-threatening disease. Most of the chest x ray datasets such as the NIH **[1]** consists of 14 different diagnosis labels which are the Cardiomegaly, Emphysema, Effusion, Hernia, Infiltration, Mass, Nodule, Atelectasis, Pneumothorax, Pleural Thickening, Pneumonia, Fibrosis, Edema and Consolidation. An additional 'No Finding' label is also added to show that there is no sign of disease.

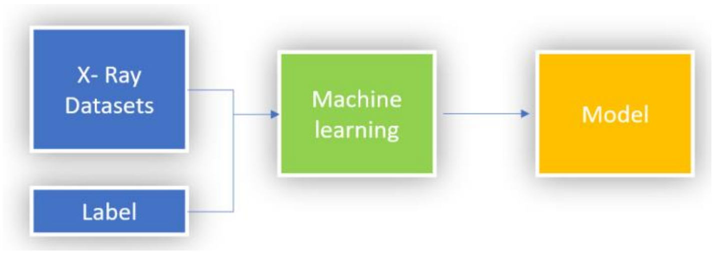


Figure 1: A brief example to demonstrate how training procedure work

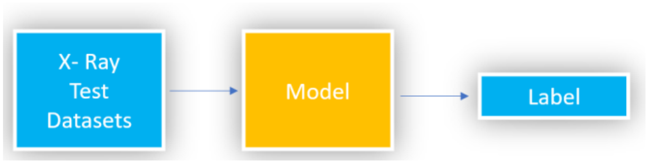


Figure 2: A brief example to demonstrate how testing procedure work

Use of AI Technologies

Overview of AI Technologies

The above figures shown just a brief demonstration on how training and testing procedure work. In this section, I will be demonstrating the use of AI Technologies more explicitly. The datasets can be consolidated into meta dataset and image dataset. Meta dataset consists of many feature variables such as the person information such as name, gender, age, the diagnosis label, path of the image shown in **Table 1** and the image dataset consists of the chest x-ray image shown in **Figure 3**.

Image ID	ID	Name	Age	Gender	Height	Weight	Labels	Image index
0012345.png	123456G	John wee	52	M	1.72	78	Mass	../input/image_dir/0012345
0012346.png	123461	Mary	44	F	1.66	55	Pneumonia	../input/image_dir/0012346
…	…	…	…	…	…	…	…	…

Table 1: An example of meta datasets in chest x-ray

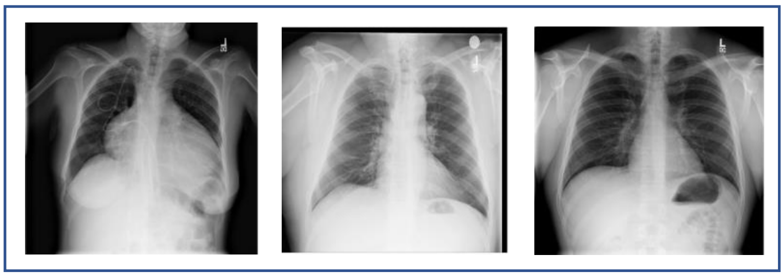


Figure 3: An example of chest x-ray image

Since this is a supervised learning task (label is provided), we train a model using meta and image datasets. The first step is to make use of the meta datasets show in **Table 1** with machine learning techniques such as decision trees, navie baye and K-NN to train a model. Personally, I prefer ensemble estimator such as random forest as it aggregates many classifiers to enhance the performance.

Random Forest is a bagging ensemble classifier that only contains decision trees. The different between bagging and boosting is how they assigned the weight to the misclassified dataset. For bagging, the weight for the misclassified dataset is evenly distributed where for boosting it adds more weight to the misclassified dataset.

The second step is to use the image datasets shown in **Figure 3** to train a model such as CNN or Transformer. The label can be extracted from the meta datasets. We can use a pre-trained Resnet-50 model from pytorch and disable the gradient on all model to freeze the weight and replace the final fully connected layers with two fully connected layer and set the learning rate to 2e-5.

The third step is to combine random forest and Resnet-50 model during inference. However, we will not use major voting for this aggregation instead we can assign a specific weight for both the estimators. But is possible to find correct assigned weight for each estimator? Indeed, it is possible. We introduce Optuna framework [2] which is a state-of-the-art hyperparameter optimization framework that search the optimize weight for both the estimators. Now both the estimators have their own optimized assigned weight shown in **Figure 4**.

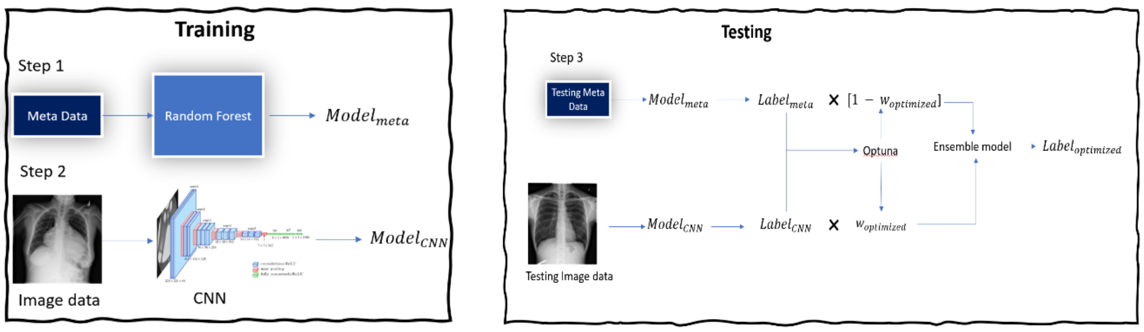


Figure 4: A diagram showing my proposed solution for training (Left) and Testing (Right)

Responsible AI Practices Addressed

Fairness

Problem

Fairness in machine learning can exist in many different forms. It is not necessarily come from imbalanced datasets. Unfairness can be still introduced even if the datasets consist of same number of datasets. This is because the model can potentially introduce bias. For instance, we train the datasets that contain same amount of chest x-ray of two diagnosis labels, Hernia and No Finding, and same amount of gender for feature variable. During testing phrase, the model nevertheless keeps predicting that female has Hernia (False positive), which also known as the evaluation bias. Another type of bias is called the Aggregation Bias where during model construction, it combined inappropriately as such resulting bias. Next, we will be looking at a few types of fairness as wrong metric can lead to harmful decisions.

Demographic parity is one of the fairness metrics that require equal proportion of positive predictions in each group. However, demographic parity is too strong and too weak. Before we discuss about the weaknesses, lets understand what demographic parity is. First, we set a threshold in training to decide who will be diagnosed with Pneumonia as shown in **Figure 5**. Let's assume the threshold is 0.5 and green dots represent Male and blue dots represent Female, the darker coloured dots represent the gender with Pneumonia. The PR for both gender is 50%. As we can see the situation is obviously quite unpleasant, but demographic parity is completely fine with it. To back up, there are papers that argue that demographic parity benefits the reputation of the smaller disadvantageous groups. [3] Therefore, I will not be using demographic parity as my fairness metrics.

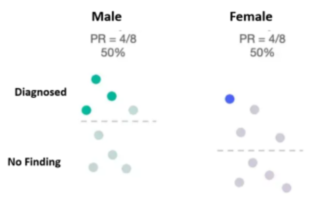


Figure 5: An example demonstrating a dataset using demographic party unfairness metric

Equalized odd is also another fairness metrics that trade false positive rate of one group for false negative rate of another group. False positive rate and false negative rate can be defined as $\frac{FP}{FP+TN}$ and $\frac{FN}{FN+TP}$ respectively. Using the similar diagram shown in **Figure 6**, the false positive rate for male is = $\frac{1}{1+3}$ and false positive rate for female is = $\frac{1}{1+3}$ which satisfies the condition P (FPR | M) = P (FPR | F). Therefore, I will be using Equalized odd for fairness metrics.

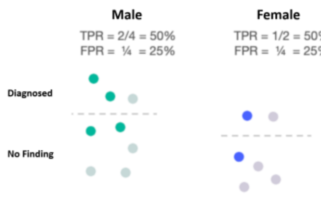


Figure 6: An example demonstrating a dataset using Equalized Odds unfairness metric

Let's assume we want to know if the model is bias toward Male or Female. So, we drop all the features in the datasets except for Gender and Label to train a model classifier without any fairness constraints.

	Accuracy		False Positive Rate		False Negative Rate	
Gender	Train	Val	Train	Val	Train	Val
F	0.688	0.682	0.314	0.2898	0.309	0.345
M	0.672	0.674	0.128	0.104	0.639	0.667

Table 2: Demonstrating an example of gender bias in dataset using threshold 0.5

Table 2 shows the accuracy, FPR and FNR for training and validation. The train and validation set accuracy of the classifiers approximate the same for Male and Female. However, the train and validation set FPR and FNR differ substantially, which indicate that Female are often predicted to being diagnosed with diseases while the Male are more often predicted as not 'No Finding'. The disparity in equalized odds (FNR) for validation can be calculated by P (FNR=Val | M) – P (FNR=Val | F) = 0.667 – 0.345 = 0.322. According to Equalized odds, both FPR and FNR must be equal for both protected and unprotected group. In this case, the FNR and FPR for both protected and unprotected group is not equal which means there is unfairness in the model. Next, we will be looking at the few solutions to solve unfairness.

Solution

There are serval approaches that can incorporate fairness constraints into machine learning model. These approaches are pre-processing, constraints learning and post processing. For pre-processing, we adjust the training data to mitigate model unfairness. Constraints learning incorporates fairness constraints in the loss function and post-processing methods make adjustment to the existing training model to satisfy the fairness constraints such as adjusting the parameters in the trained model. However, I will be focusing on post-processing methods mainly finetuning the decision threshold using ROC.

In the previous example the threshold is set to 0.5 and to debias the model, we need to choose a separate threshold for each sensitive groups. Now, we will select the new decision threshold to ensure that the new FPR for Female must be equal to the new FPR of Male. In **Table 2**, the train FPR for Male is 0.13. Therefore, we will be using 0.13 as a target FPR for Male.

Next, we will input the train label and the predicted train label using ROC curve to determine a list of new FPR, TPR and threshold. Then, we argmin the subtraction of the new FPR with 0.13. The purpose of this is to find the index of the smallest value in the new threshold.

In coding, we can write it as:

```
Target_FPR = 0.13
new_FPR, new_TPR, new_threshold = roc_curve(train_label, train_pred)
min_idx = (new_FPR – Target_FPR).argmin()
new_thres = new_threshold[min_idx]
```

So, now let's assume the new threshold is 0.6. We retrained the model using 0.6 threshold shown in **Table 3**.

	Accuracy		False Positive Rate		False Negative Rate	
Gender	Train	Val	Train	Val	Train	Val
F	0.622	0.633	0.130	0.111	0.597	0.613
M	0.672	0.677	0.128	0.104	0.639	0.633

Table 3: Demonstrating an example of gender bias in dataset using the new threshold 0.6

The disparity in equalized odds (FNR) for validation can be calculated by P (FNR=Val | M) – P (FNR=Val | F) = 0.633 – 0.613 = 0.02 and the train and validation set accuracy of the classifiers approximate the same for Male and Female.

As can be seen from this table, we are able to substantially reduce the disparity in equalized odds (FNR) for validation from 0.322 to 0.02. Even though the mean validation accuracy drops slightly from 0.68 to 0.65, we have successfully satisfied the condition for equalized odds with a small ε. This method is then applied to all the protected groups such as age, race to ensure that the model is fair. For instance, we also can incorporate the same ideas to CNN.

Private Preservation

Problem

As a chief of Medical Officer, you need to ensure that the privacy of the patient is priority, and all data are confidential which cannot be release to data scientist directly to train a model shown in **Figure 7**. Even if there are confidentiality agreement, are we able to trust them? We have no control over the data, we do not know who is accessing it, will the data be misused and is there third party involve? So, how can we allow data scientist to train a model without intruding the privacy of the patient and protect the data even if it leaks?

User
Data
Data Scientists

Figure 7: A diagram demonstrating the procedure of the user releasing confidential data to Data Scientists without applying any privacy preservation

Solution

For this problem, I proposed SVD-federated learning to prevent any third parties from retrieving model directly and to protect any leakage of model during aggregation process. SVD (Singular Value Decomposition) is one of the principal component analysis (PCA) that stored the most important feature while reducing the dimension of the data. SVD can be defined as a m x n ***M*** matrix that is a factorization of the form of $U\Sigma V^T$where U is the m x m matrix, Σ is a m x n rectangle matrix with non-zero on the diagonal side and V is the n x n matrix. Imagine we have a meta dataset ***M*** of dimensions 9000 x 9 shown in Figure 8. We can factorize it into a form of $U\Sigma V^T$. The data are now split into three different sub matrices. However, we will focus on federated learning first.

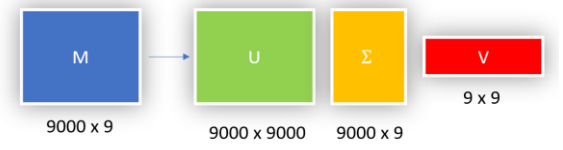


Figure 8: A diagram demonstrating a data of 9000 x 9 is converted to a form of U, Σ and V^Tusing SVD

Federated learning can be defined as to train AI models without anyone seeing or touching the data. This means that instead of transferring local data to the server, federated learning transfer model to local data. The following Figure 9 will illustrate the procedure of the federated learning in medical imagining.

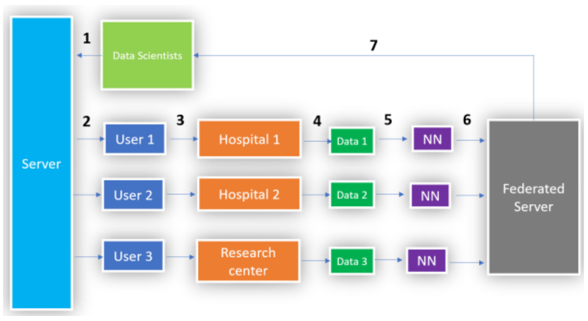


Figure 9: A diagram demonstrating the architecture of privacy preservation using federated learning

1) The data scientist will send a notification for approval and upload the global model (Neural Network) to the users via the server.
2) The users received the notification and decided whether to accept or decline
3) If the users accept, it will notify the hospital or the research centre
4) To approve the data and get ready for training
5) The users need to download the uploaded model and each endpoint will incrementally train a global model based on the local data
6) After training, the users upload the weight instead of the data to the centralized location (Data scientist)
7) The data scientist received the models and aggregate it and analysis the performance via many metrics such as accuracy, recall, precision and F1 score. This process will repeat until the model is robust.

However, there is an issue in step 7. The aggregated update is not protected from the users and if someone happen to disconnect all the training will go to waste. To add on, this also give the model a chance to have seen all the different kind of data and using generative adversarial networks (GAN) we have a chance that the model will reconstruct the data. Hence, the common method is to add noise to trade-off some of the accuracy to the model.

For my solution, I proposed SVD federated learning to factorize the local model into a form of U, Σ and V^T and store every of them into a separate sub-model shown in Figure 10 before uploading the model to the federated server. The model can be retrieved by reconstruction. Even if there is model leakage, the model is useless without any reconstruction. For example, one day one of the users upload the local model to federated server, and it happens that the V sub-model leaks. We do not need to worry because V sub-model is useless without the reconstruction of U and Σ sub-model. Moreover, if the eigenvalue of the last rank is 0 or close to 0 in Σ matrix, we can remove the entire columns since the last columns of the eigenvalues has the least significance to the model. However, I want to incorporate an additional parameter α after the reconstruction of sub-model of U, Σ and V^T to further protect the model from being misused. α is a random variable that is added into the sub-model.

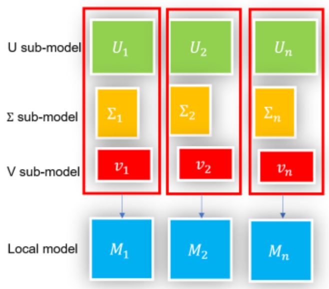


Figure 10: A diagram that demonstrate the sub-model of every component is append into a list of that sub-model.

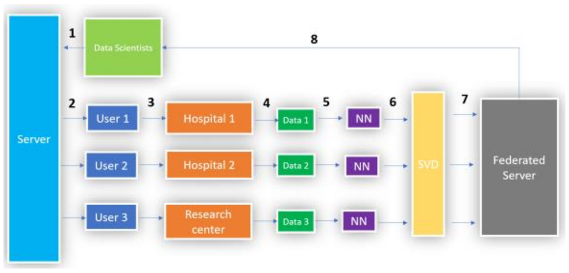


Figure 11: A diagram demonstrating the architecture of privacy preservation using SVD-federated learning

Figure 11 shows the final architecture of SVD-federated learning. Step 1 to step 5 remains the same. For step 6, instead of sending the local model to the federated server, we use SVD to factorize the local model into a form of U, Σ and V^T and store every of them into a separate sub-model. This method is to prevent any lost information to be misused by hacker when uploading it to the federated server. In step 7, the sub-models will reconstruct back to the local model and add some random variable so that no one can inspect the global model in the federated server. For multiple users, we can group them into a group of 5 and assign the same α. This way it will reduce the computational memory and all the data will be safely protected.

Explainability

Problem

We can train a robust model that have an accuracy of 90% but if there is no trust between human and machine learning it is useless. Human need to know how machine learning make assumption such as decision trees. But from the diagram in lecture notes, we have seen that the neutral network has the highest accuracy but low interpretation. Moreover, how can we convince experts in another domain that the prediction make by the machine learning is correct? For example, if we train a classifier and predict the chest-x ray has the diagnosis label of Pneumonia but the doctor disagrees and based on his experience he thinks is Edema. In this situation, who is correct? In the following section, I will explain how to solve the issue in interpretation of machine learning and neural network.

Solution

Normally, the standard machine learning procedure is to insert input data, choose the machine learning algorithm, and predict the test label. However, to incorporate interpretability in machine learning, we need to use SHAP (Shapley Additive Explanation) shown in **Figure 12** or add another layer for neural network shown in **Figure 13.**

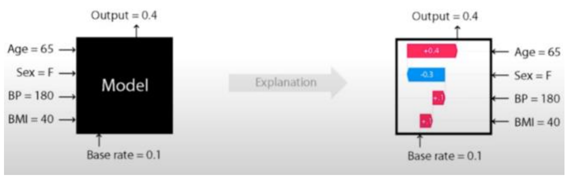


Figure 12: A diagram that demonstrate data input into a black box

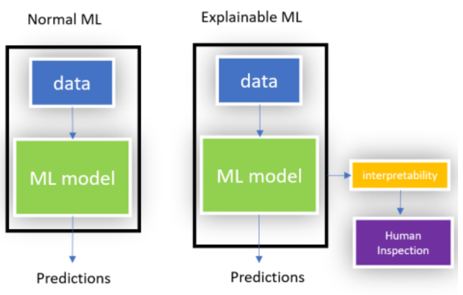


Figure 13: A diagram that demonstrate the difference between with (Right) and without (Left) the incorporated interpretability in the model

Meta – data

There are two datasets – meta and image. For meta, we can use SHAP to determine how important the feature contribute to the predicted label. Let's imagine we have 2 labels – 'No Finding' and Edema First, we need to convert categorical data into binary 0 and 1. Since there is only one Euclidean distance, we do not need to convert the label to one-hot encoding. We do the same for gender.

Image ID	ID	Name	Age	Gender	Height	Weight	Labels	Image index
0015233.png	15233X	Jane Ng	23	F	1.66	45	?	../input/image_dir/0015233.png

Table 4: An example of a test dataset without labels

For example, in **Table 4**, we use random forest, and predict the testing labels as '0'. But why random forest predicts 0 and not 1? To solve this, we need to figure out how each of the features such as age, gender, height, and weight, contribute to the output label. For instance, we are interested to know which feature contribute the most.

For demonstration, I will be focusing on the 'weight' feature. But how does the weight 45 kg contribute to the output label? To solve this, we can analysis shapely value for 'weight' by fixing all the features in test data and set a random value just for weight. These features are then input into the model to predict the label shown in **Table 5**. Next, we find the mean absolute error between the testing label and the random labels to get the shapley value. In this case the shapely value for weight is 0.25, which is less than 0.5, so the prediction is 0. The next step is to apply these methods to find all the shapely value for the remaining features. As we can see, the combination is 2^n which is computational expensive.

Age	Gender	Height	Weight	Labels	MAE
23	1	1.66	56	1	0 -1 = 1
23	1	1.66	42	0	0 - 0 = 0
23	1	1.66	41	0	0 -0 = 0
23	1	1.66	48	0	0 - 0 = 0
					Mean MAE= 0.25

Table 5: An example of a dataset that fix all the features in the test data and randomize the weight feature to find the Shaply value

Another alternative solution is to use random permutation on the features. Initially the features are arranged in form of [Age, Gender, Height, Weight], we can randomize the sequence of the features to [Height, Gender, Weight, Age] and generate random sample, Random data 1 shown in **Table 6**. Then we formulate vector X1 with testing label of [Height, Gender, and Weight] with Random data 1 of [age] to make prediction. We formulate another vector X2. This time containing testing label of [Height, Gender] with Random data 1 of [Weight, Age] to make prediction. The local shaply value φ_1 can be computed by subtracting X1 from X2. We repeat this procedure for many times and sum them together and find the mean of the shapely weight φ_i which is equivalent to SHAP (w=45| [23,1,1.66,45]) = ***E*** (Difference).

Example 1:

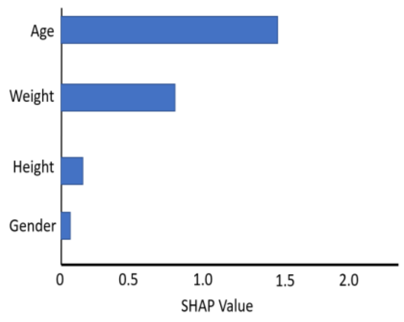
Random data 1:

Height	Gender	Weight	Age
1.62	0	59	25

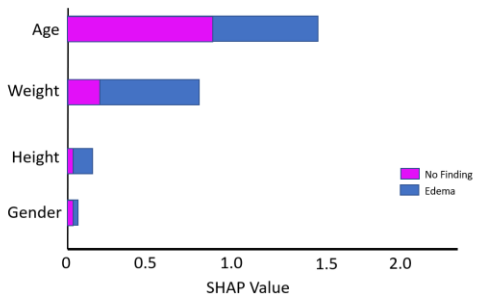
	Height	Gender	Weight	Age	Labels
X1:	1.66	1	45	25	1
X2:	1.66	1	59	25	1
					Diff = 1-1 = 0

Table 6: (Top) A random sample with different feature sequence (below) X1: Fix the Height, Gender and Age using testing datasets and use random sample's weight. X2: Fix the Height and Gender using testing datasets and use random sample's weight and age to predict a label and find the MAE between X1 and X2 to get the shapely value.

After you have found all the shaply values for all features, we can plot a bar chart shown in **Figure 14**. For instance, the most important feature shown in **Figure 14a** is Age and the least important feature is gender and in **Figure 14b** the bar chart include labels for each feature. For instance, Age feature has 55% (purple) of 'No Finding' while 45% (Blue) 'Edema'.



(a)



(b)

Figure 14: (Left) is sorted by the important features and on the right is further classified by the label

Therefore, we can apply SHAP in our machine learning algorithm to interpret the most important feature to clients.

Image data

For image dataset, I proposed Grad-CAM [4] also known as Gradient Class Activation Mapping that compute the gradient of the target label in the last feature maps to produce a localization map that highlight the important regions in the images.

In CNN, it can be break down into three components. In the first components it contains convolutional layers and pooling layers which concentrate on feature extraction. In the second component, the last feature map is flattened into a linear layer which also known as fully connected layer that concentrate on feature classification. The last layer, SoftMax, convert the weight to probability.

For instance, to understand the concept of Grad CAM, we need to understand how feature maps are generated.Feature maps are generated by applying kernel or filters to the input image show in **Figure 15.** For example, an input image of 5 x 5 and the kernel size is 3 x 3 shown in **Figure 15.** The feature maps can be achieved by applying the kernel to the image pixels and using max pooling to down sample the dimension. The kernel act like a sliding window. Assume the stride is 1, the kernel will start at the red box, then move to green, then yellow and subsequently the whole image.

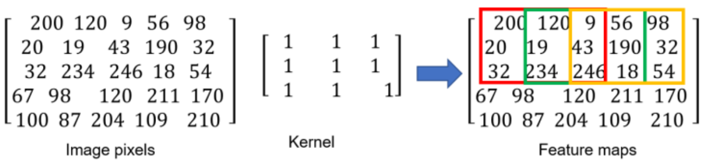


Figure 15: An example of a feature maps by applying kernel to the image pixels

For Grad-Cam it uses the last feature map to creates the heat map as it contains the spatial and critical information for that predicted label. Now we will look at how every pixel in the input image contribute to the feature maps.

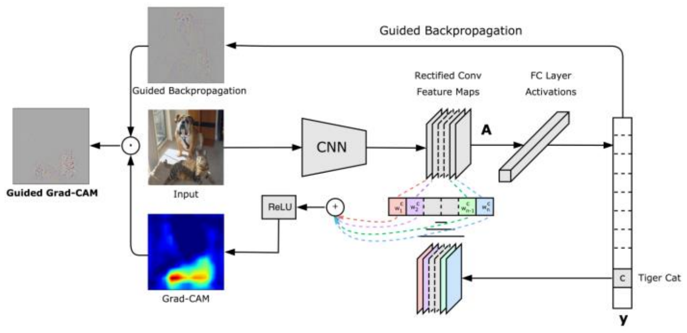


Figure 16: A diagram that show the architecture of Grad-CAM

1) First, we compute the gradient of the diagnosis label with respect to $A_{i,j}^{K}$, where A^k is the last feature map and *i* and *j* is the row and columns.
2) Then we average the sum of the gradient score a_k^c to find the weight value of each pixel in the feature maps shown below, where Z is the total number of pixels (width x height) and a_k^c represent the weight value for feature maps

$$\alpha_k^c = \frac{1}{Z}\sum_i\sum_j \frac{\partial y^c}{\partial A_{ij}^k}$$

3) Next, to find CAM (Class Activation mapping) we multiple the weight a_k^c to the entire feature maps $A_{i,j}^{K}$ to look at how each of the initial pixels in the image contributed to the specific class. Then, we add ReLU activation function to remove all negative values.

$$L_{\text{Grad-CAM}}^c = ReLU\left(\sum_k \alpha_k^c A^k\right)$$

4) Lastly, we need to swap the last layer SoftMax to a linear layer to compute Grad-CAM

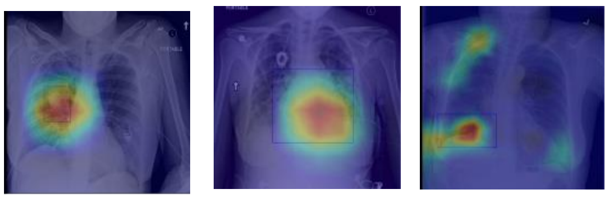


Figure 17: Example of a few chest x-ray datasets using Grad-CAM and heatmap for interpretation

From **Figure 17** we can see that the heatmap highlighted the location of the disease. For instance, red colour indicates stronger response. Therefore, it is easier for other domain experts (doctors) to interpret what CNN focusing on in the chest x-ray.

Final Design

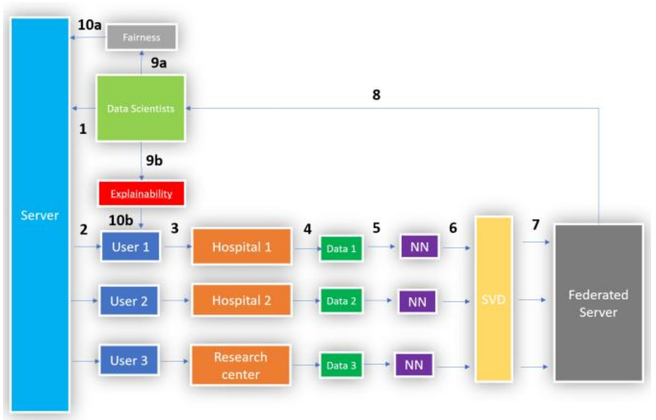


Figure 18: A combination of all the Responsible AI that includes Privacy Preservation, Fairness and Explainability

In conclusion, I incorporated all the factors in Responsible AI into an architecture. Step 1 to 8 remain the same. For step 9a and 10a, I incorporated Fairness in the system. After we aggregate the model, data scientist needs to test if the data is fair. If the result shows bias (FPR is not equal to FNR), he will adjust the threshold using ROC and retrain the model until it satisfied the equalized odds condition. For 9b and 10b, we assume the data is already fair and data scientist will apply SHAP or Grad-CAM to the model for better visualization and it is also easier for domain expert to understand the most important feature in the dataset.

Reference

[1] https://www.kaggle.com/datasets/nih-chest-xrays/data
[2] Optuna: A Next-generation Hyperparameter Optimization Framework. Takuya Akiba1, Shotaro Sano, Toshihiko Yanase. 2019
[3] A Short-term Intervention for Long-term Fairness in the Labor Market. Lily Hu, Yiling Chen. 2018.
[4] Visual Explanations from Deep Networks via Gradient-based Localization. Ramprasaath R. Selvaraju Abhishek Das Ramakrishna Vedantam. 2016.