Deep Learning Application in Medical Image

RSNA Intracranial Hemorrhage Detection

2019-11-25

1 Project Overview

Deep Learning techniques have recently been widely used for medical image analysis, which has shown encouraging results especially for large healthcare and medical image datasets. In the computer vision field, the deep learning model, such as Convolutional Neural Network(CNN) has shown better capabilities to segment and/or classify medical images like ultrasound and CT scan images in comparison to traditional machine learning techniques.

Recently, Deep Learning applications, in particular in applying the CNN model for analyzing Medical Images have achieved very promising results. The major application fields can be broadly separated into two categories: classification application and segmentation applications.

- Classification Applications

For a given set of labeled images, using the deep learning model to find the patterns between the input images and its corresponding class labels. The related applications, such as lung images detection from CT scanning to classify images patches into 7 classes[1]. This paper describes how to use the CNN model to classify the healthy tissue and six different interstitial lung disease patterns. The other example is to identify the thyroid nodules as malignant or benign from the chest X-ray and Ultrasound images[2].

- Segmentation Application

The other important application for Medical Image Analysis is to identify organs, lesions or substructures of organs from the Ultrasound, MRI or X-Ray images. Now, you can use deep learning models to segment the brain tumors from MRI images.[3]

With recent progress in Deep Learning field, this project will build a model and application to detect acute intracranial hemorrhage and its subtypes based on the rich medical image dataset which is provided by the Radiological Society of North America (RSNA®) in collaboration with members of the American Society of Neuroradiology and MD.ai. This is also a Kaggle Featured Prediction Competition launched months ago.[4]

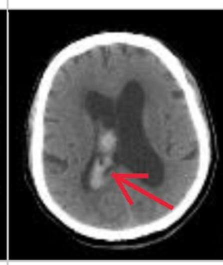
[1] "Lung Pattern Classification for Interstitial Lung Diseases Using ...." https://ieeexplore.ieee.org/iel7/42/7463083/07422082.pdf.
[2] "Classification of thyroid nodules in ultrasound images using ...." https://ieeexplore.ieee.org/document/7952290.
[3] "Brain Tumor Segmentation - Papers With Code." https://paperswithcode.com/task/brain-tumor-segmentation.
[4] "RSNA Intracranial Hemorrhage Detection." https://www.kaggle.com/c/rsna-intracranial-hemorrhage-detection/overview/description.

1.1 Problem Statement

Intracranial hemorrhage, bleeding that occurs inside the cranium, is a serious health problem requiring rapid and often intensive medical treatment. For example, intracranial hemorrhages account for approximately 10% of strokes in the U.S., where stroke is the fifth-leading cause of death. Identifying the location and type of any hemorrhage present is a critical step in treating the patient.

Diagnosis requires an urgent procedure. When a patient shows acute neurological symptoms such as severe headache or loss of consciousness, highly trained specialists review medical images of the patient’s cranium to look for the presence, location and type of hemorrhage. The process is complicated and often time consuming.

This project is to develop a Classification/Segmentation model and build a web application to identify the five Hemorrhage sub-Types: Intraparenchymal, Intraventricular, Subarachnoid, Subdural and Epidural.

	Intraparenchymal	**Intraventricular**	**Subarachnoid**	**Subdural**	**Epidural**
Location	Inside of the brain	Inside of the ventricle	Between the arachnoid and the pia mater	Between the Dura and the arachnoid	Between the dura and the skull
Imaging					
Mechanism	High blood pressure, trauma, arteriovenous malformation, tumor, etc	Can be associated with both intraparenchymal and subarachnoid hemorrhages	Rupture of aneurysms or arteriovenous malformations or trauma	Trauma	Trauma or after surgery
Source	Arterial or venous	Arterial or venous	Predominantly arterial	Venous (bridging veins)	Arterial
Shape	Typically rounded	Conforms to ventricular shape	Tracks along the sulci and fissures	Crescent	Lentiform
Presentation	Acute (sudden onset of headache, nausea, vomiting)	Acute (sudden onset of headache, nausea, vomiting)	Acute (worst headache of life)	May be insidious (worsening headache)	Acute (skull fracture and altered mental status)

1.2 Metrics

- This project are are evaluated using a weighted multi-label logarithmic loss. Each hemorrhage sub-type is its own row for every image, and the model will predict a probability for that sub-type of hemorrhage. There is also an ‘`any`’ label, which indicates that a hemorrhage of

ANY kind exists in the image. The **'any'** label is weighted more highly than specific hemorrhage sub-types.

- For each image `Id`, prediction will have a set of predicted probabilities (a separate row for each sub-type). Then taking the log loss[5] for each predicted probability versus its true label. The ***loss*** is averaged across all samples.
- When calculating a weighted multi-label logarithmic loss, predicted input values of 0 and 1 are undefined. To avoid this problem, log loss functions typically adjust the predicted probabilities (p) by a small value (epsilon) and use the MinMax Rule: $max(min(p,\ 1-10^{-15}),\ 10^{-15})$

1.3 Development Framework

In Deep learning application field, there are two major deep learning frameworks: TensorFlow and PyTorch. The competitive strengths for each framework are:

- TensorFlow is mainly adopted by the industrial companies and PyTorch is mainly focused on research communities.
- TensorFlow has a large, well established user base, and industry is typically slower to pick up on new technologies. TensorFlow is much more efficient than PyTorch. Even modest savings in model run times can help a company's bottom line.
- PyTorch integrates neatly with Python, making the code simple to use and easy to debug.

In this project, we will use the Keras to implemtation the model.

1.4 Dataset

The original Dataset is hosted on the Kaggle platform[6]. The download API :
kaggle competitions download -c rsna-intracranial-hemorrhage-detection

There are two-part data:

- Train.csv: include the ID and Label:
  ID is a combined string that includes the image filename and Hemorrhage type.
  Label is a target column, indicating the probability of whether that type of hemorrhage exists in the indicated image.
  Format:
  `[Image Id]_[Sub-type_Name]`, as follows:
  ```
  Id,Label
  1_epidural_hemorrhage,0
  1_intraparenchymal_hemorrhage,0
  1_intraventricular_hemorrhage,0
  1_subarachnoid_hemorrhage,0.6
  1_subdural_hemorrhage,0
  1_any,0.9
  ```
- DICOM Images:

[5] "What is Log Loss? | Kaggle." https://www.kaggle.com/dansbecker/what-is-log-loss.
[6] "RSNA Intracranial Hemorrhage Detection | Kaggle." https://www.kaggle.com/c/rsna-intracranial-hemorrhage-detection/data.

DICOM is the standard for the communication and management of medical imaging information and related data. DICOM files can be exchanged between two entities that are capable of receiving image and patient data in DICOM format.
DICOM images contain associated metadata. This will include `PatientID`, `StudyInstanceUID`, `SeriesInstanceUID`, and other features.

2. Project Theoretical workflow

Workflow is very important in machine learning project. this project's will follow the following workflows and pipelines:

- **Data Exploration and Analysis**
  Understanding the dataset, Create a meta dataset from DICOM image Dataset
  Read DICOM image and analysis image data, Windowing Scaling, Normalizing Dataset

- **Pre-processing Data**
  Remove useless images information, resample the dataset to a small dataset to quick prototyping, crop images that only include important information, e.g. only have brain tissues, rescaling into 256*256 px image and reduce the huge data size for training

- **Data Refinement and Data Augmentation**
  Apply the data augmentation methods to improve the model generalization and reduce the overfitting.

- **Model Evaluation and Validation**
  Train and Valid dataset split, use the pretrained Model to train and fine tune,optimization

- **Model Justification and Benchmark Comparison**
  This project will choose the `Resnet50` as the baseline model. Will compare each model's performance with the above benchmarks.

3. Data Exploration and Analysis

RNSA Dataset provide the rich DICOM images (Almost 400GB DICOM images) and training information data. Before to build the model, it is key to understand the images and data. The Data analysis Jupyter notebook can be found here
(https://github.com/Pyligent/RSNA-Medical-Image-Detection/blob/master/RSNA%20Intracranial%20Hemorrhage%20-Data%20Exploration%20.ipynb)

3.1 Data Exploration

Stage_1_train.csv(Stage 1) file includes all labels information, which include the subtype and id string. Total shape of this csv is (4045572, 2). Please see figure 1 to see the Labels value counts.

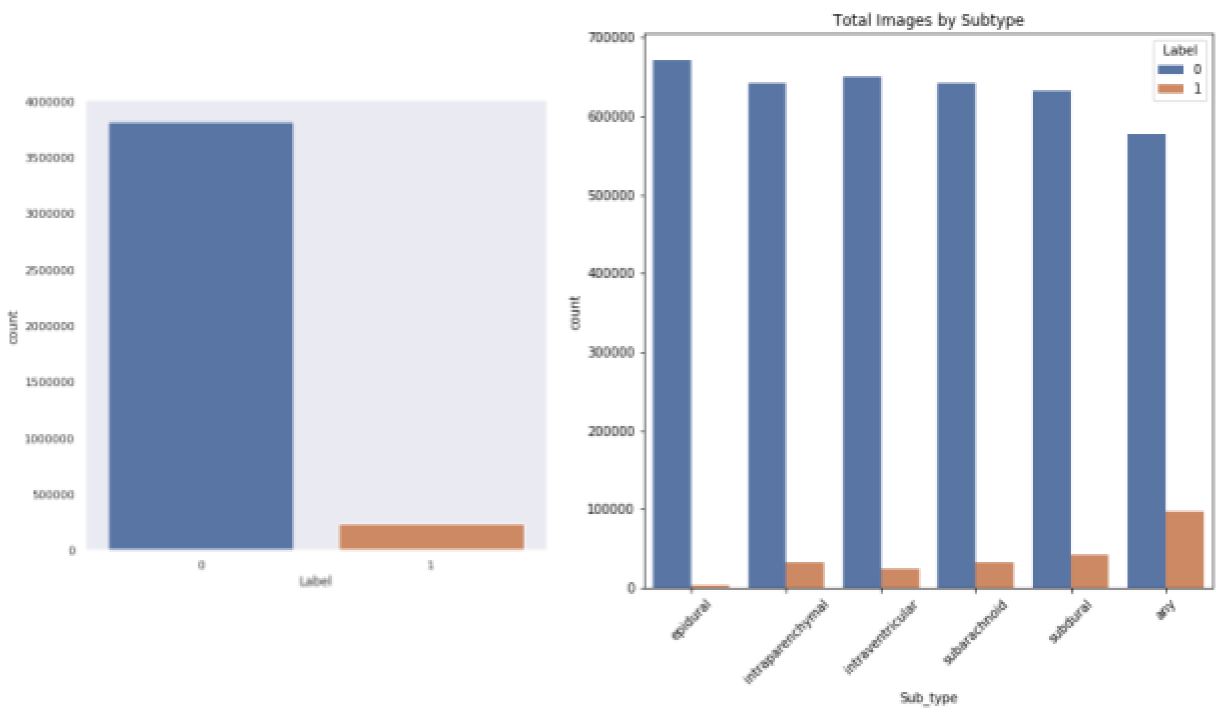


Figure 1: Label Information Plot

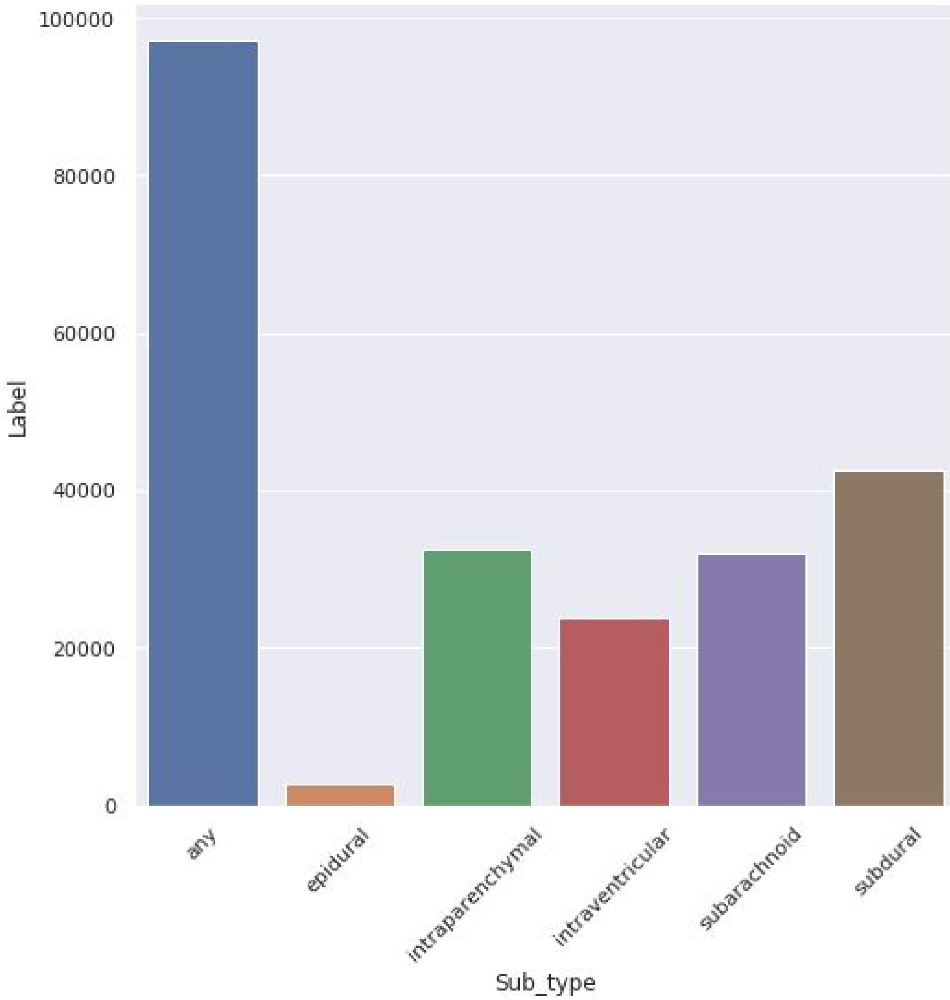
Obviously, the dataset is imbalance. (Label:0 ,3814760 v.s Label: 1 ,230812) , the balance rate is only around 6%.The Imbalance Problem is a common problem affecting machine learning due to having disproportionate number of class instances in practice. The easy way to deal with the imbalance dataset is to apply the sampling based approaches.
There are three methods in sampling:

- Oversampling: to add more of the minority class so it has more effect on the machine learning algorithm
- Undersampling: to remove some of the majority class so it has less effect on the machine learning algorithm
- Hybrid, a mix of oversampling and undersampling

In this project, will combine the undersampling and oversampling approaches and fine-tune the model to achieve better results.

Label Types

There are six sub-type in the label dataset: 'any', 'epidural', 'intraparenchymal', 'intraventricular', 'subarachnoid', 'subdural' . The following chart shows the label's distribution.



3.2 Data Visualization

Understanding the DICOM images is important in this project. For medical image, the windowing image will let model only focus on the useful information and also will let us to resize and reduce the image dataset, so it will be easier to train the model.
By using the PYDICOM library, I have built the the function the show the metadata and display different sub-type dicom images.
The detailed Jupyter notebook can be found at project's Github[7].

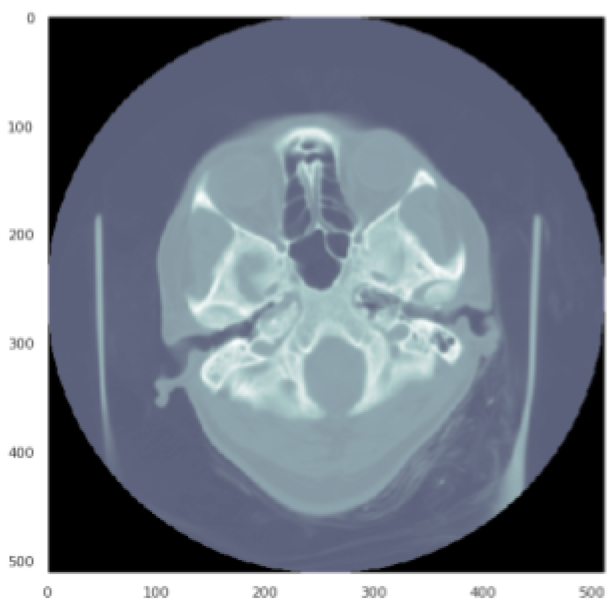
```
def show_dicom_metadata(filename):
    """
```

[7]https://github.com/Pyligent/RSNA-Medical-Image-Detection/blob/master/RSNA%20Intracranial%20Hemorrhage%20-Data%20Exploration%20.ipynb

```
    show_dicom_metadata function is to get all important DICOM metadata, such as
windowing parameters and other information and also plot it
    input parameter:
    filename: string, DICOM filename
    """

def display_dicom_image(df, sub_type, column_number,row_number):
    """
    display_dicom_image function shows the DICOM image from the training dataset
dataframe.
    df: data frame that includes the images and subtype information
    sub_type: string, what sub_type want to show
    column_number: int, how many images in a row
    row_number: int, how many rows want to show
   """
```

```
show_dicom_metadata(train_imgs[0])
```

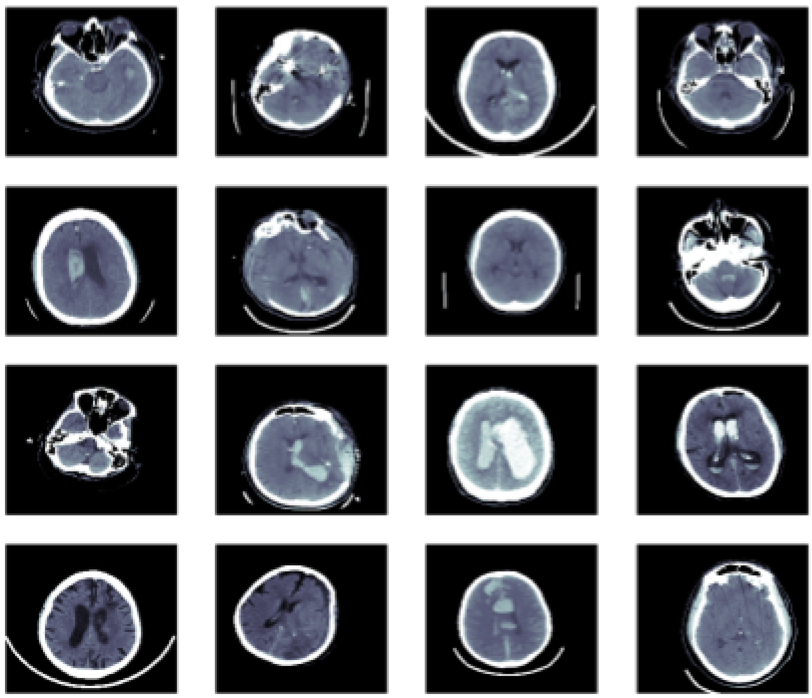
```
Filename.........: ../input/rsna-intracranial-hemorrhage-detection/stage_1_train_images/ID_000039fa0.dcm
Storage type.....: (0008, 0018) SOP Instance UID                    UI: ID_000039fa0
Patient id.......: ID_eeaf99e7
Modality.........: CT

Window Center.........: 30
Window Width.........: 80
Rescale Intercept.........: -1024
Rescale Slope.........: 1
Image size.......: 512 x 512, 524288 bytes
Pixel spacing....: ['0.488281', '0.488281']
Slice location...: (missing)
```



Sub_Type: intraventricular

```
display_dicom_image(train_df, 'intraventricular', 4,4)
```

Images of Hemorrhage Sub-type:intraventricular



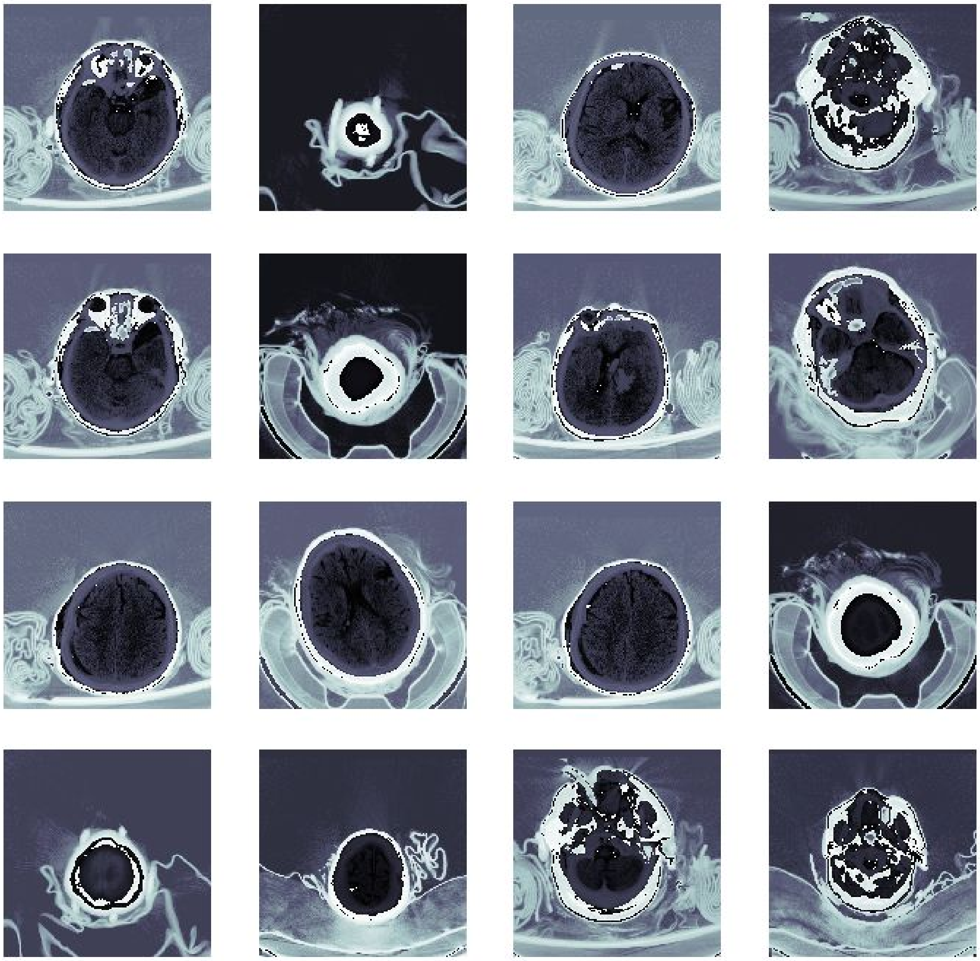
3.3 Data Analysis and Preprocessing

The raw medical image need carefully analysis and preprocess before input into the deep learning model. Some images have a lot useless information/noise or without any information. Also we need extract and windowing the right image zone from raw DICOM images, then we can accelerate the training process and increase the accuracy.

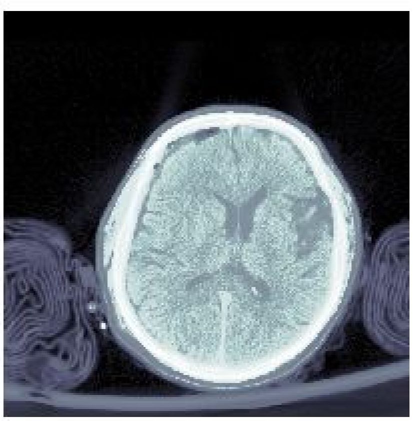
In this project, we will use the Fastai Medical Image libraries to analysis and raw DICOM images

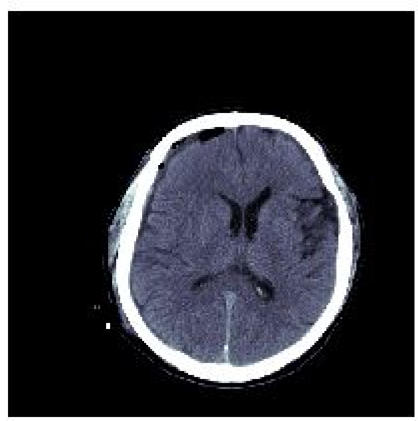
The following steps are how to analysis and preprocess the DICOM images:

1. Check the PixelRepresentation values in raw DICOM:
   `PixelRepresentation == 0` , there's *normally* a `RescaleIntercept` of `-1024` to give us signed data, the max `RescaleIntercept` for that row is `1.0`.
   `PixelRepresentation == 1`, `RescaleIntercept` is normally also `-1024`, even although it's signed data, which means that shouldn't be necessary.
2. Choose RescaleIntercept > -1000

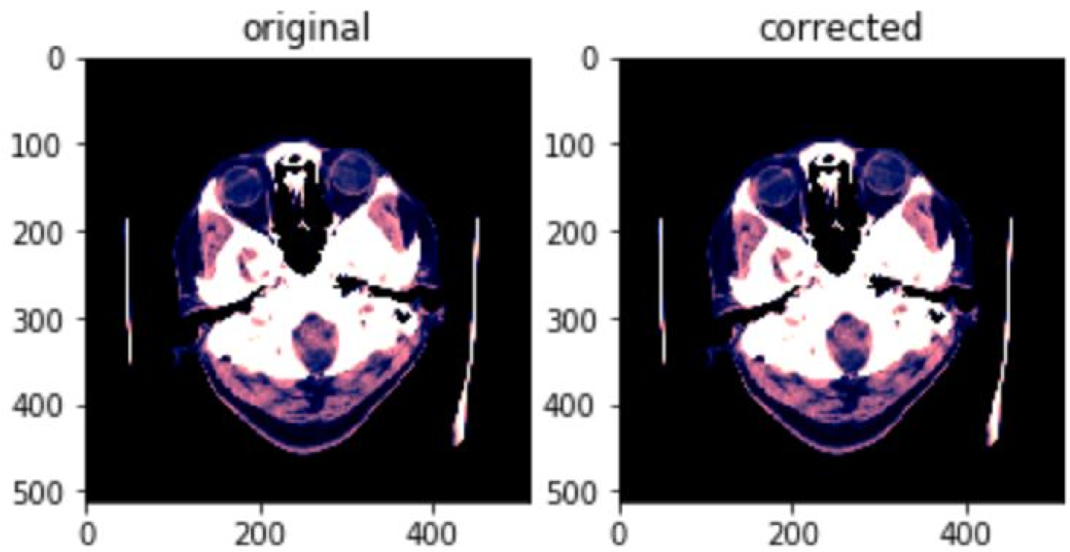


3. Plot the hidogram of Pixels and remove the part of value == 0





4. Windowing the brain Area



The complete notebook link is https://www.kaggle.com/ansojin/clean-data.

4 Data Pre-processing, Refinement and Augmentation

4.1 **Data Pre-processing and Refinement**

After analysis the DICOM image, we will build the right dataset to feed model. The following steps will be foillowed:

- Based on the PixelRepresentation issues in DICOM image, build the helper function to process the DICOM image
- Build the windowing and extract function to extract the brain tissue part from DICOM
- Resize the right size to feed the model

4.2 **Data Generationor and Imbalance Handling**

Build the DataGenerator Class to produce the Train and Test Data sets. One of the key problem for the DataSet Generator is to handle the dataset unbalance problem. In this project, will try both oversampliing and undersampling methods for model training.

- ResNet50 Model:
  Use the undersampling technical to keep the probabilities of the label 'any' value balance
- EfficientNet B4 Model:
  Use the oversampliing technical to over-sample images of the label value = 'epidural'

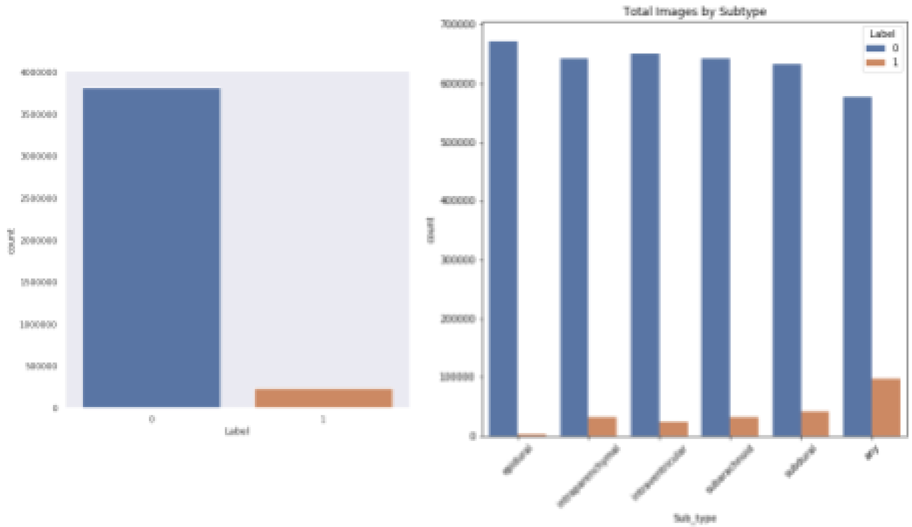


Figure 1: Label Information Plot

4.3 Data Augmentation

Data augmentation is a strategy that enables deep learning algorithm to significantly increase the diversity of data available for training models, without actually collecting new data. Data augmentation techniques such as cropping, padding, and horizontal flipping are commonly used to train large neural networks. However, most approaches used in training neural networks only use basic types of augmentation. While neural network architectures have been investigated in depth, less focus has been put into discovering strong types of data augmentation and data augmentation policies that capture data invariances.So in this notebook will use the random transforms which apply flip, warp, rotate,zoom,lighting and contrast randomly.

Also in this project, will try different way to train the model.

- ResNet50 Model: Without the data augmentation
- EfficientNet B4 Model: Will use the imgaug library for image augmentation.
  The parameters setting:
  1. The augmentation effects
     decrease the probability of some augmenters to be applied by decreasing in : sometimes = lambda aug: iaa.Sometimes(0.25, aug)
  2. Use the augmentation sequence to randomize the augmentation process
     Horizontal Flip rate: 0.25
     Verical Flip rate: 0.10
     AdditiveGaussianNoise : per channel 0.5

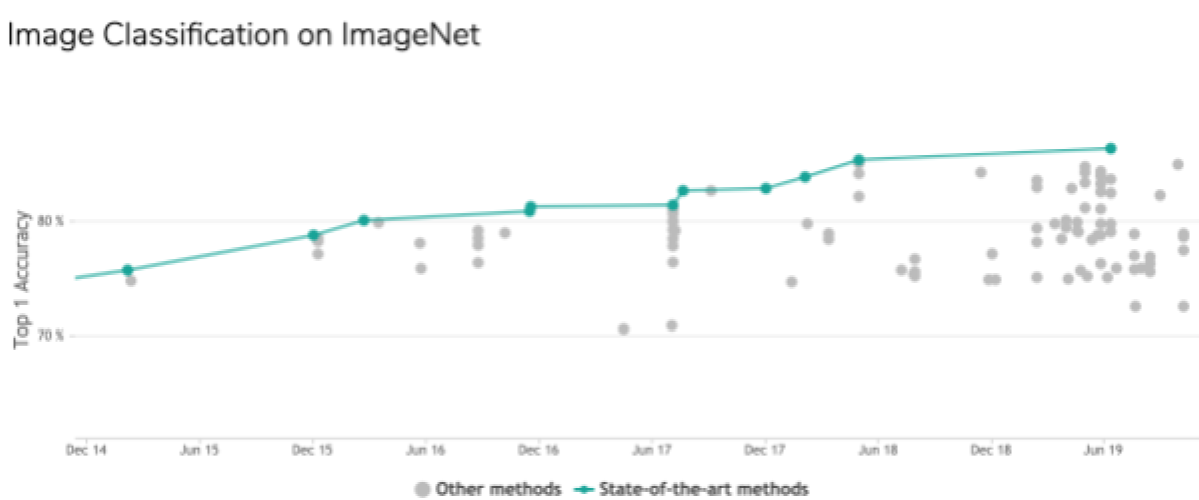
GaussianBlur: sigma = (0, 0.5)
ContrastNormalization : 0.75 - 1.5
Crop
3. Random order for each training image

5 Model Evaluation and Validation

5.1 State of the Art Models

Medical Image Classification is a core problem in computer vision and deep learning R&D area. How to choose the right model is the key for the projects. Usually, the model choosing will be based on the benchmark on the ImageNet database. The models pre-trained for image classification, and then transfer to a variety of computer vision applications. This project will follow this methodology and use the transfer learning method to apply the pre-trained models to build the hemorrhage subtype detector.

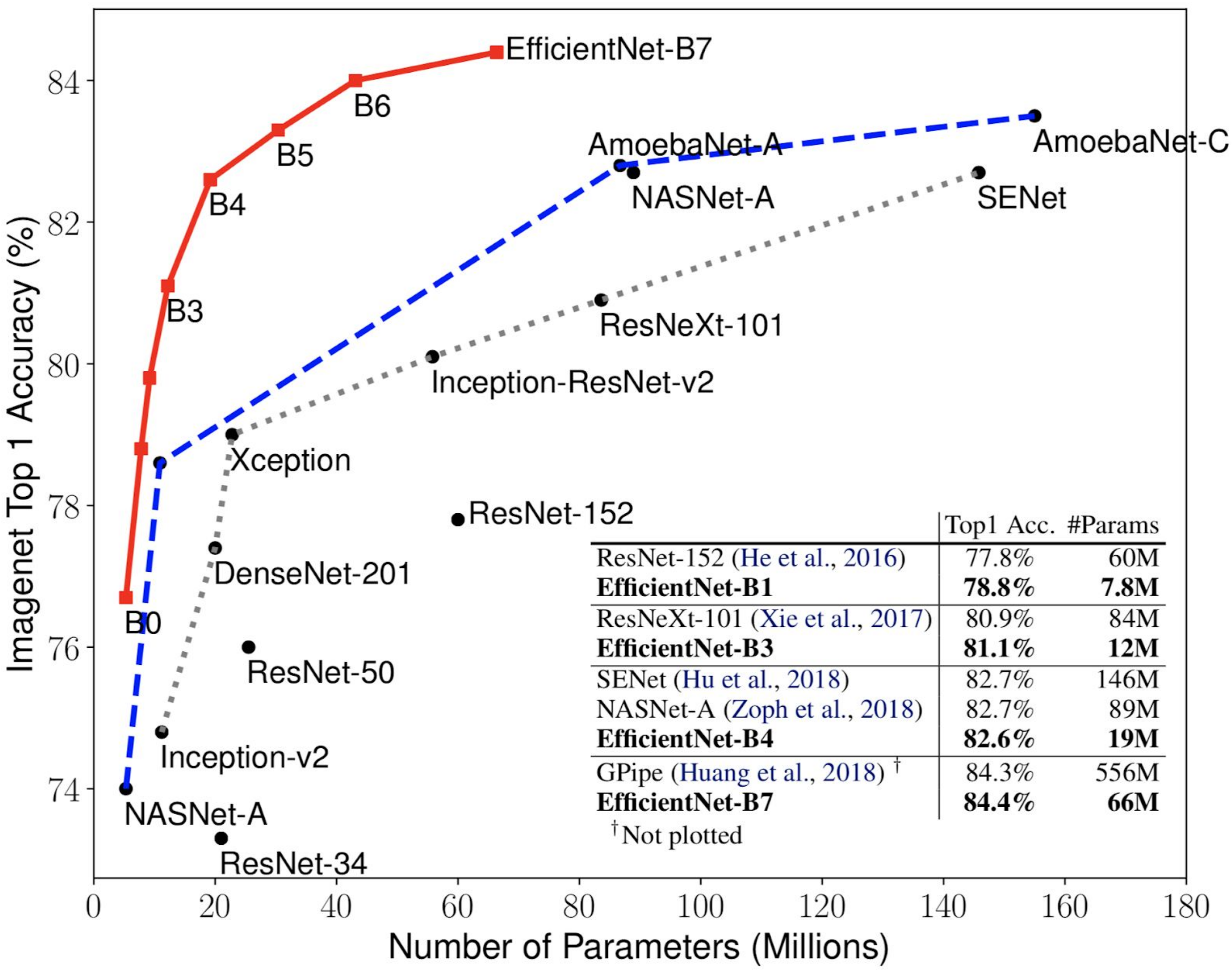
Recent research in image classification has demonstrated improved performance by using the larger neural networks and higher images. For example, the State of the Art(SoTA) model in benchmark is currently held by ResNeXt-101 32*48d architecture[8] with 829M parameters by using 224*224 images for training. More efficiently SoTA Model is EfficientNet-b7[9] with just 66M parameters by using 600*600 images for training.More recently, revised ResNeXt-101 32*48d models has got 86.4% accuracy[10].
More detailed information of SoTA models is as follows[11]:



[8] "Exploring the Limits of Weakly Supervised Pretraining." 2 May. 2018, https://arxiv.org/abs/1805.00932.
[9] "EfficientNet: Rethinking Model Scaling for Convolutional ...." 28 May. 2019, https://arxiv.org/abs/1905.11946.
[10] "Fixing the train-test resolution discrepancy." 14 Jun. 2019, https://arxiv.org/abs/1906.06423.
[11] "Image Classification on ImageNet - Papers With Code." https://paperswithcode.com/sota/image-classification-on-imagenet.

5.2 **Model and Network Architecture**

For medical image analysis, Convolution Neural Network models have achieved the great progresses. But the network archtecture of the CNN models are becoming more deeper and more complicated.



Due to huge Dataset (+400GB), this project has to train the model only on the Kaggle platform. But the 30hrs/per week GPU limitation is big hurdles in the model training cycles and model debuging.
So, among all those models, this project will choose the moderate network archtectures to train the model. Pretrained ResNet50 and newly published EfficientNet B4 have the good results on ImageNet and the paramters also below the 30M. So this project will use ResNet 50 and EfficeintNet B4 model to build the hemorrhage subtype classification.

5.3 **Model Evaluation and Validation**

5.3.1 **ResNet50 Model**

Train Data Input: Data Set generator will transform the DICOM image into size 224*224 and channel 3 image as imput
Train/Valid/Test Data Split: Use the ShuffleSplit to split the train dataset
Model Hyper-parameters:

- batch size = 32
- epochs = 5
- Decay_rate = 0.8
- Learning rate = 5e-4

Validation Scores:
Public Scores: 1% test data: 0.40581
Private Scores : 99% test data: 0.07895

Training Script:
https://github.com/Pyligent/RSNA-Medical-Image-Detection/blob/master/RSNA_ResNet50_model.py

5.3.1 **EfficientNet B4 Model**

Train Data Input: Data Set generator will transform the DICOM image into size 256*256 and channel 3 image as imput
Train/Valid/Test Data Split: Use the MultilabelStratifiedShuffleSplit to split the train dataset
Model Hyper-parameters:

- batch size = 16
- epochs = 6
- drop_out = 0.2
- Learning rate = 1e-4

Validation Scores:
Public Scores: 1% test data: 0.73422
Private Scores : 99% test data: 0.17564

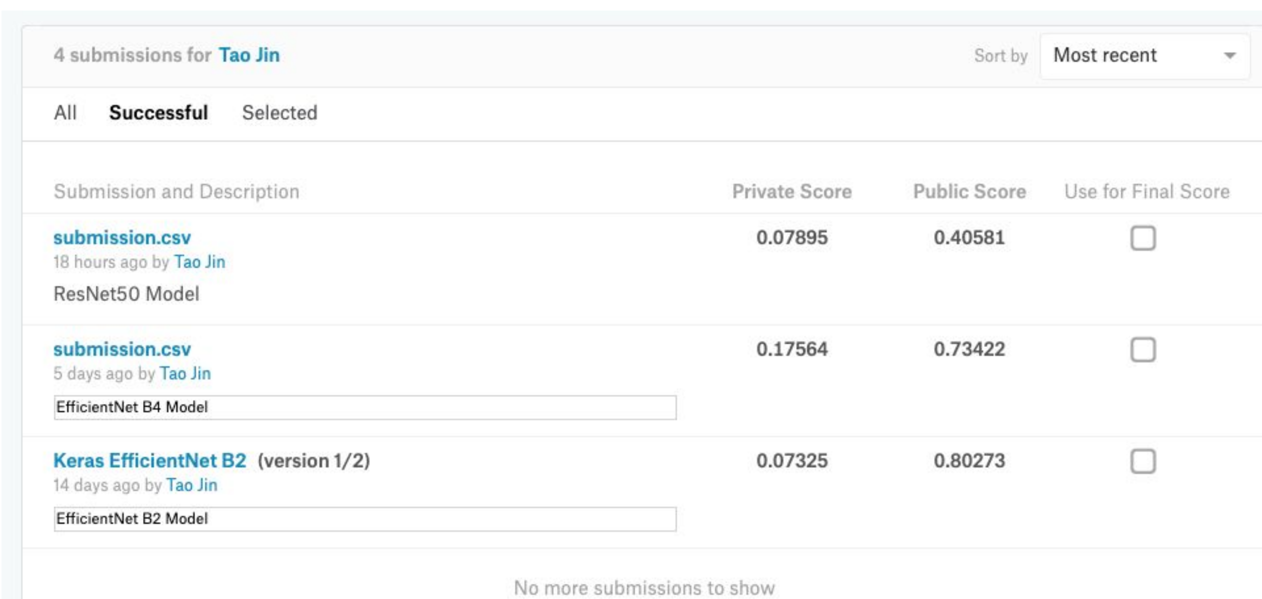
Training Script:
https://github.com/Pyligent/RSNA-Medical-Image-Detection/blob/master/RSNA_EfficientNet_B4.py

6 Model Justification and Benchmark Comparison

6.1 **Benchmark and Result Comparsion**

The benchmarks will choose the most used models in this problems.
The popular models in this problems are ReNeXt-101, EfficicentNet B0/B2 and DenseNet. My results and benchmarks omparations is in the following charts.

Models	Training Image Size	Metrics(>99% Test case)
ResNeXt-101 32x16d	256x256	0.086
EfficientNet B0	256x256	0.102
EfficientNet B2	224x224	0.080
DenseNet	224x224	0.139
My Results:		
Resnet50	**224x224**	**0.079**
EfficientNet B4	**256x256**	**0.176**



6.2 **Results Analysis and further work**

The major problem for this project is the training environment limitation. Without a dedicated GPU workstation in house, I have to train the whole model on the Kaggle platform . A full cycle training (6 epochs) always need 15+ hours without commiting. So for Kaggle's rule 30 hrs/per week, it is very hard for me to debug and tune the hyper-parameters. Also it is kind of impossible to move the 400+G data from Kaggle to GCP or AWS, it takes long time and very expensive.
Still for those two models, there are still some areas need to fine tune to achieve the good results.

1. ResNet50 Model:
    - Add data augmentations during the training
    - Try Test Time Augmentation(TTA)
    - More epoches training
    - Learning rate, decay rate still need fine tune
2. EfficicentNet Model:
    - Data Augmentation Parameters still need explore
    - Try Test Time Augmentation(TTA)
    - More epoches training
    - Hyper-parameters tuning
    - Input shape adjust 112*112 or 224*224

Further work will be more focus on the hyper-parameters tuning. The other method is to use Fastai2 medical image package, but training time for stage 2 images dataset will always exceed 30 hours. So I need create a small dataset to try out.