POLITECNICO DI TORINO

Master's Degree in Computer Engineering

Master's Degree Thesis

Comparison of self-supervised and ImageNet pretraining for medical image classification

Supervisors

Prof. FABRIZIO LAMBERTI

Dr. LIA MORRA

Candidate

HAICHAO SONG

session December 2020

Summary

Deep learning has witnessed a huge amount of attention over the last decade.It has been widely used in the field of medical images, especialy the deep neural networks for classification, detection and segmentation task. However, when compared with natural images, medical images are very difficult to label. In most cases, medical image datasets are not as large as natural image datasets.To address this problem, the traditional approach is through the use of transfer learning methods, by using ImageNet pre-trained models. But medical images are often very different from natural images, in particular, medical images have different intensities, and the images contain internal structures of the human body that are completely absent from the natural image data set.

In the field of computer vision, the ImageNet is widely used for pre-training because it is large enough (more than 14 million tagged images) that 1.2 million images divided into 1,000 categories are useful for training a general purpose model. We prefer to pre-train some general-purpose models, like ImageNet, specifically for downstream medical image processing, so that helps with downstream model training.

In this thesis, for 21 datasets, we mixed images from different modalities (CT, X-RAY, MRI) and used a self-supervised learning method during the pre-training process, applying the pre-trained weights to downstream medical image datasets for classification task. In this thesis, we transfered this pre-trained weights on LUNA16 dataset and MURA dataset, for the LUNA16 dataset, we perform a false positive reduction task on the patch level and an image classification task on the image level for the MRUA dataset. The model is trained with Resnet50 and VGG16 backbone respectively.

This thesis trained three different models, training from scratch, transfer learning by using self-supervised pre-trained weights, transfer learning by using ImageNet weights, and analyzing the differences between the three models by comparing the AUC (Area Under the Curve). We conclude that when using the VGG16 backbone for the image level classification task on MURA dataset, the AUC of transfer learning using our pre-trained model can get a similar result as transfer learning by using ImageNet. In the other three experiments, the AUC was also higher than

that of the training from scratch through transfer learning of the pre-trained model. Pre-training using a self-supervised learning approach can improve the accuracy of downstream model classification tasks.

Acknowledgements

I would like to express my sincere gratitude to my professors, classmates, colleagues, friends, and family members for their support and guidance during my three years master study.

First of all, I would like to thank my supervisor, Prof. Fabrizio Lamberti, for accepting me and giving me the opportunity to participate in this project.

I would like to thank my supervisor, Prof. Lia Morra, for her careful guidance in the selection of the topic, the design of the proposal, and the completion of the thesis, and for her patience and thoroughness in the entire process. she is very helpful to me over the year, always responding quickly to my questions, even during the night.

Thanks to Dr. Sina and my colleague Giovanni, we worked together to build this project from scratch, and with their help, I learned a lot.

I would also like to thank all of my friends who have helped me over the past three years, especially Pakman Tung, Wang Zi, and Dr. Xie Chen for their encouragement and companionship every time I encountered difficulties.

Finally, I would like to thank my family for their support and encouragement over the years, which enabled me to successfully complete my studies, it is their care and dedication that created superior conditions for me and made me optimistic and courageous in my life.

Table of Contents

List of Figures

Chapter 1

Introduction

In recent years, machine learning and deep learning have been used with great success in medical imaging. Researchers have used deep learning methods in tasks such as medical image segmentation, detection, and classification with fairly good results. Despite the tremendous advantages of using these methods, there are some drawbacks that need to be addressed. Medical images are difficult to obtain image labeling information, which requires doctors and radiology staff to label the images. In addition, there are so many complicated disease categories, with sub-categories within the larger categories, it is easy to have an imbalance in labeling.

In the past, there were different approaches to deal with this problem, such as transfer learning, semi-supervised and self-supervised learning, which have also achieved good results. The use of pre-trained models on ImageNet is one of the most used, but for medical images, their imaging principles are completely different from those of natural images, medical images have different intensities, and the images contain internal structures of the human body that are completely absent from the natural image data set. Therefore, the effect of transfer learning from natural images to medical images is not as pronounced as that of transfer learning between natural images. To address this problem Zhongwei Zhou et al[1]. propose "Model Genesis", which is a new approach, self-supervised learning to solve the problem of the lack of a large number of labeling information in medical images, in their paper, they made some changes on the original image, and then let the model to restore the original image. In this way, the original image itself becomes a label for supervised model training, consistent with the original intent of self-supervised learninglearning features directly from the data.

However,the approach they used was to train on 3D images and to train separate models for each of the different modalities (CT,X-RAY), the idea in this thesis is whether the Genesis model can be better realized if all modalities (CT, X-RAY, MRI) of medical images are mixed together for training? Also the models were trained in 2D, Maimone Giovanni have also added more pre-text tasks to make the

model more generic.

We pre-trained 21 datasets containing different diseases, different body parts, and different modalities, and transfer the pre-trained weights to the downstream LUNA16 dataset and the MURA dataset for classification tasks.l.The experimental results show that using our pre-trained model for transfer learning yields better results than a model trained from scratch and the state of the art ImageNet transfer(MURA using VGG16).

1.1 Thesis structure

1. Chapter 1 : An introduction to the thesis
2. Chapter 2 : Background and Related works,The rest of the document is structured as follows: a detailed description of the theoretical knowledge of the techniques used in this thesis, as well as the previous research in the field.
3. Chapter 3 : Materials and methods. In this chapter we provide a detailed description of the datasets used in the thesis, as well as the preprocessing and sampling strategies for the datasets, and introduce the two strategies used in the paper[2]: batch normalization and instance normalization
4. Chapter 4 : Materials and method.In this section, I describe in detail the pre-training model and the specific implementation of transfer learning on the LUNA16 dataset.
5. Chapter 5 : Conclusions and future works

Chapter 2

Background and Related works

2.1 Background

Nowadays, deep learning has made great achievements in the field of computer vision.The use of deep learning(DL) has been increasing rapidly in the medical imaging field, including computer-aided diagnosis (CAD), radiomics, and medical image analysis.The following sections will describe in detail the main techniques used in this thesis.

2.1.1 An introduction to deep learning

Deep learning(DL), a branch of machine learning, is an algorithm that attempts to perform high-level abstractions of data using multiple layers that contain complex structures or consist of multiple nonlinear transformations. Deep learning is a machine learning algorithm based on learning the representation of data, there are several deep learning frameworks, such as convolutional neural networks and recurrent neural networks have been used in computer vision, speech recognition, natural language processing, audio recognition and bioinformatics and other fields and obtained excellent results.

In this thesis, we primarily use convolutional neural networks as a deep learning framework.

2.1.1.1 Convolutional neural network(CNN)

Before the advent of convolutional neural networks, image processing was a problem for Artificial Intelligence for two reasons: 1) the amount of data to be processed

was too large, resulting in high cost and low efficiency. 2) the problems of computer version before CNNs were more related to the difficulties in engineering good and generalizable features that digitization.

The first problem CNNs solve is "simplifying complex problems" by downscaling a large number of parameters to a small number of parameters and then processing them. What's more: in most of our scenarios, dimensionality reduction doesn't affect the result. For example, if a 1000-pixel image is reduced to 200 pixels, it doesn't affect whether the naked eye recognizes a cat or a dog in the image, and neither does the machine!

Deep learning is largely inspired by human perception of images. For different objects, human vision performs cognition by grading layer by layer.

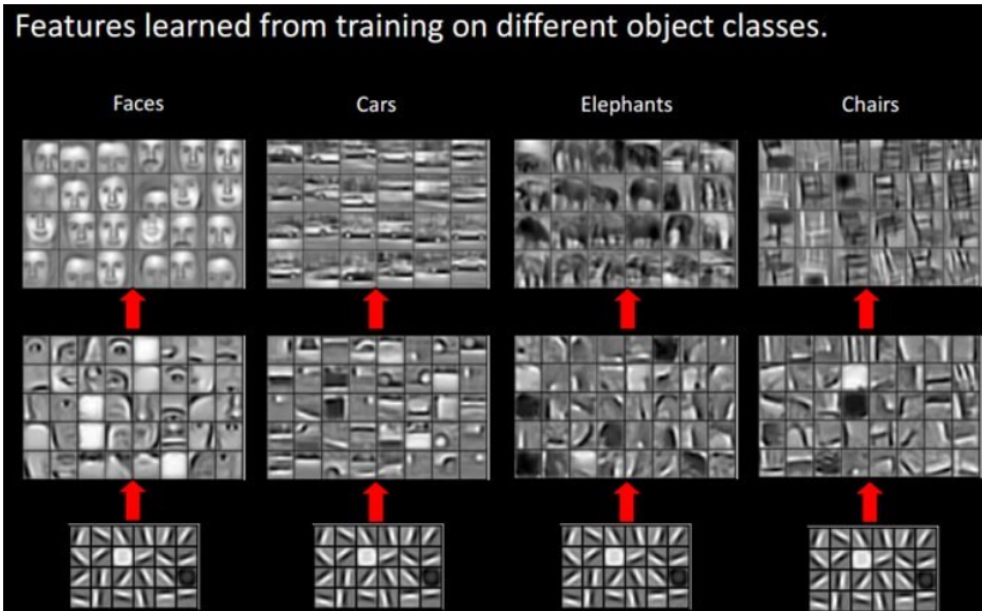

Figure 2.1: Features learned from training on different object classes

At the lowest level features are essentially similar, i.e., various edges, and the higher you go, the more you can extract some features of such objects (wheels, eyes, torso, etc.), and at the top level, the different high-level features are eventually combined into corresponding images that allow humans to accurately distinguish between different objects.

2.1.1.2 Convolutional layer

The convolution layer is responsible for extracting local features from the image. The image features are extracted by setting and moving the convolution kernel to

generate new pixel value, as shown in Figures 2.2. Center element of the kernel is placed over the source pixel.The source pixel is then replaced with a weighted sum of itself and nearby pixel.

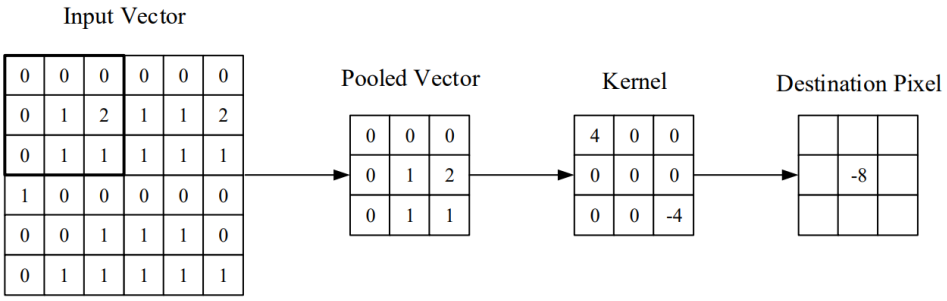

Figure 2.2: Convolutional layer [3]

2.1.1.3 Pooling layer

The pooling layer can reduce the dimensionality of the data more effectively than the convolutional layer, which not only reduces the computational effort, but also avoids overfitting. Figure 2.3 depicts how the pooling layer works.

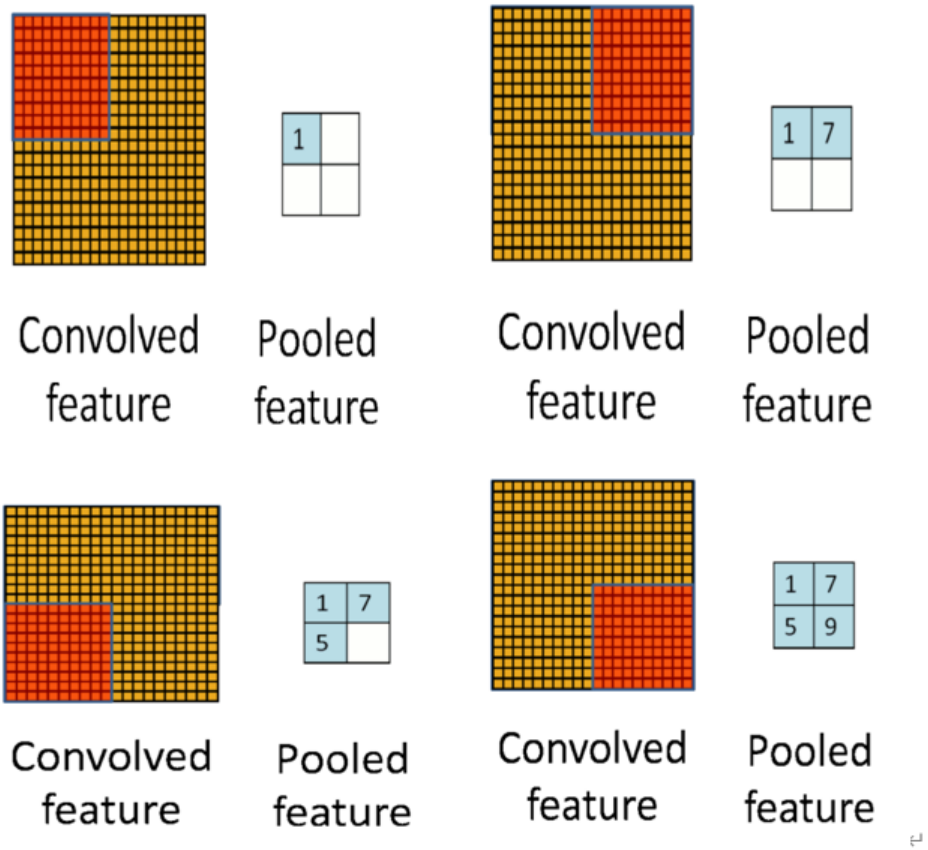

Figure 2.3: Pooling layer

2.1.1.4 Fully-connected layer

The data processed by the convolutional and pooling layers is fed into the full connection layer to get the final result. After the convolutional and pooling layer downsampled data, the full connection layer can "run", otherwise the amount of data is too large, computationally costly and inefficient. Figure 2.4 depicts how the fully-connected layer works

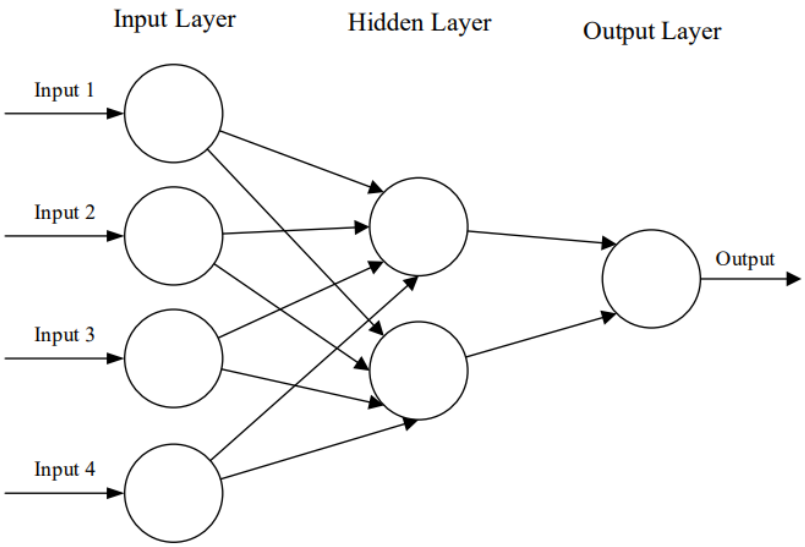

Figure 2.4: full connected layer[3]

A typical CNN is not just a 3-layers structure as mentioned above, but a multi-layers structure, such as LeNet-5, as shown in the Figure 2.5. Convolutional layer->Pooling layer->Convolutional layer->Pooling layer->Convolutional layer->Full connected layer

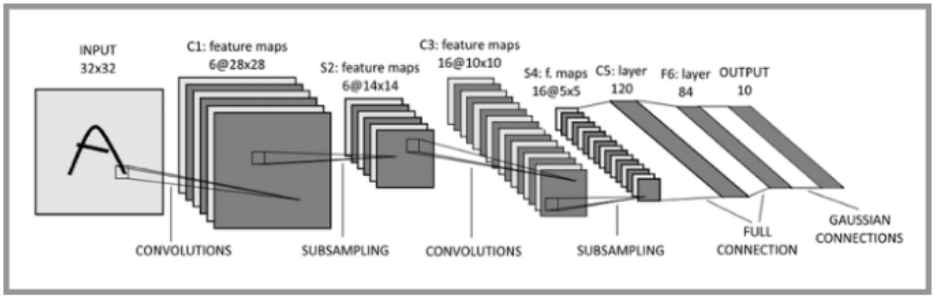

Figure 2.5: typical CNN structure[4]

2.1.2 Deep learning basic algorithms

In this section, this thesis focus on three common algorithms for deep learning, and in this thesis mainly focus on the self-supervised method.

1. supervised learning: Supervised learning is a type of machine learning task. It derives predictive functions from labeled training data. Labeled training

data means that each training instance includes inputs and desired outputs. In conclusion: given the data, the prediction labels.

2.unsupervised learning: Unsupervised learning is a type of machine learning task. It infers conclusions from unlabeled training data. The best example of unsupervised learning is cluster analysis, which can be used in the exploratory data analysis phase to discover hidden patterns or to group data. In conclusion: given the data, look for hidden structures.

3.self-supervised learning: Self-supervised learning is a subset of unsupervised learning methods. Self-supervised learning refers to the learning method in which ConvNets is explicitly trained with "pseudo" labels,this is called pre-text task, where features can be transferred to several different computer vision tasks.

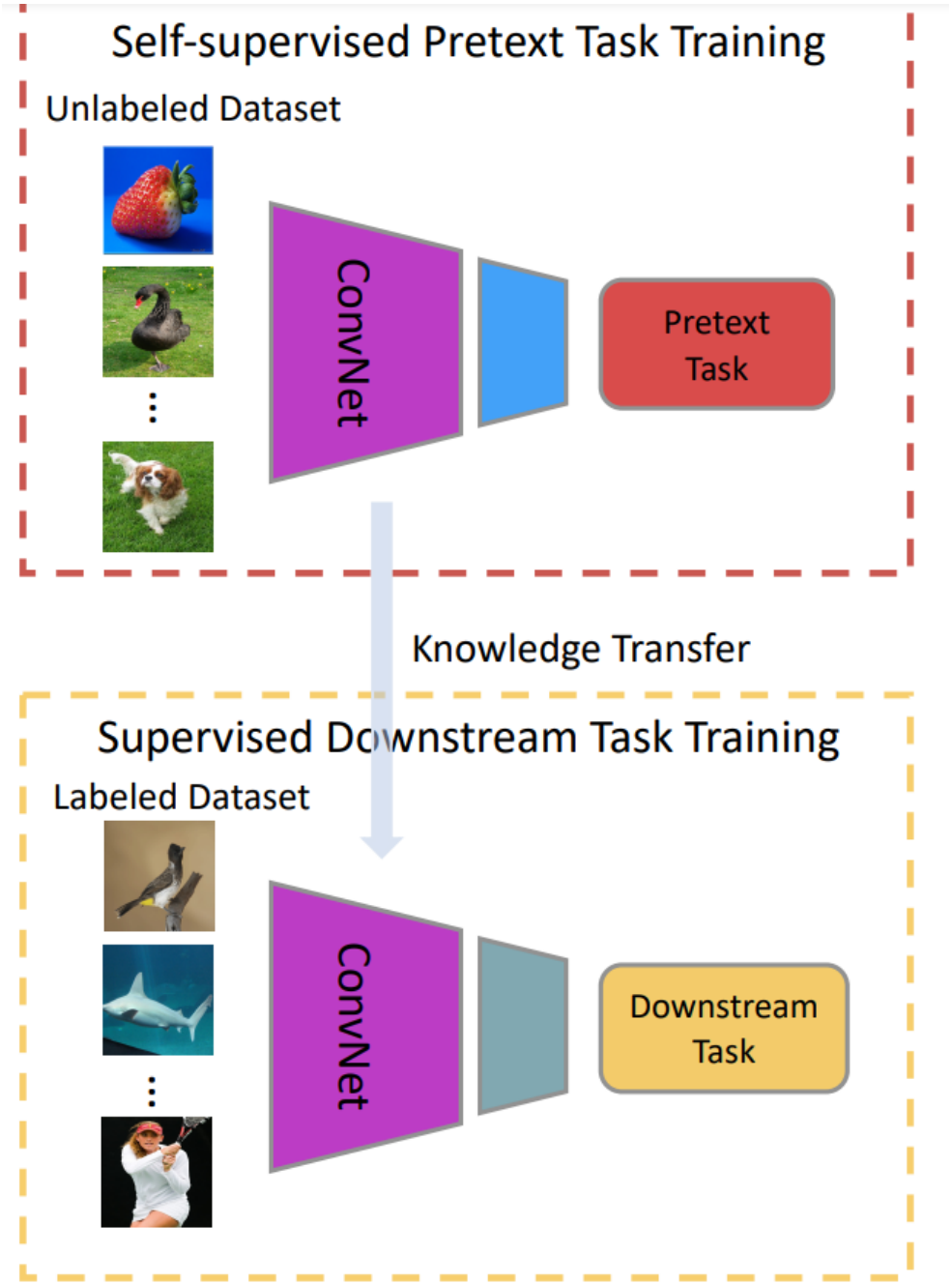

Figure 2.6: The general pipeline of self-supervised learning[5]

As it shows on Figure 2.6, the feature is learned through the pre-defined "pretext task", and after finish the pre-text task training, there will be downstream supervised

learning training through a knowledge transfer learning.

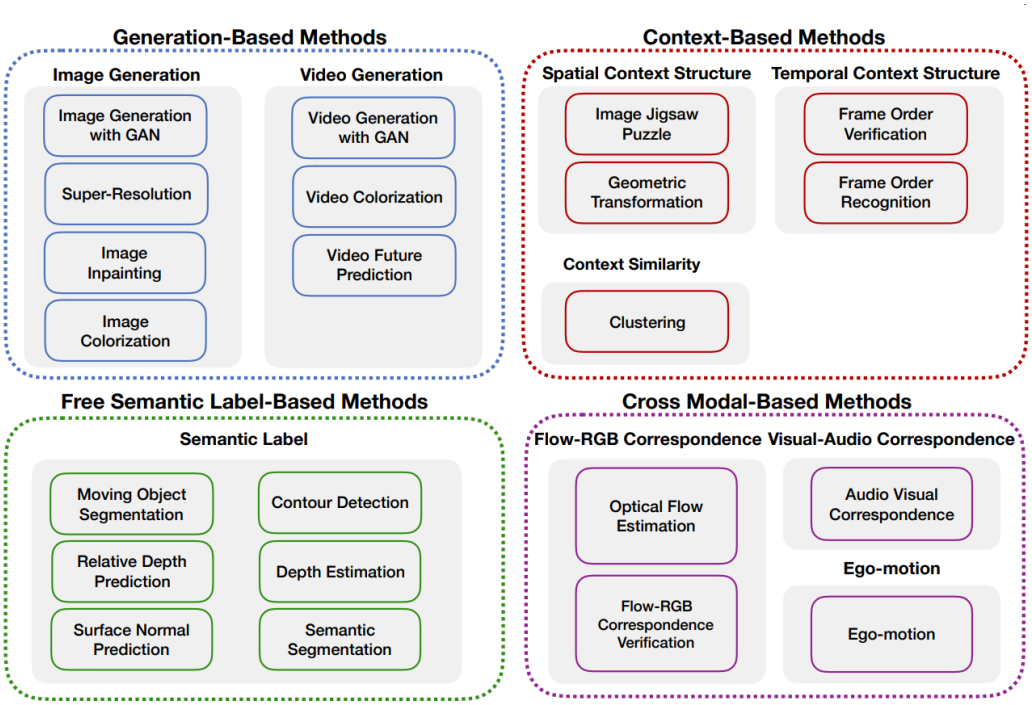

Figure 2.7: Categories of pretext tasks for self-supervised feature learning[5]

According to the data attributes used to design pretext tasks, as shown in Fig. 2.7, we summarize the pretext tasks into four categories: generation-based, context-based, free semantic label-based, and cross modal-based[5].

2.1.3 Transfer learning

The two features that are important for a deep learning model to be very effective are data and structure. Structural optimizations can be adjusted by studying the characteristics of backbone and neck with a lower threshold, but the threshold and cost of augmenting the model from a data perspective is much higher. In order to obtain a model that is generalizable and accurate, a large amount of labeled data is often required, but what can be done when it is not available in a short period of time? This is the time to use the idea of transfer learning.

One of the common uses of transfer learning is to take a pre-trained image classification model and transform it into a target detection model or a key point regression model. The reason for this is that the image classification model can be trained with the image classification dataset, and it is easier to obtain a large

number of image classification datasets, such as the Imagenet, while the number of images in the target detection dataset is much smaller and the number of samples in the key point regression dataset is even smaller, so if the image classification model is trained directly with these small number of images without transfer learning, the effect will not reach the desired accuracy and may cause overfitting.

We want the network to find the right weights for multiple iterations of forward and backward propagation by using pre-trained models that have been previously trained on large datasets, we can apply them directly to the problem we are facing, using the appropriate structure and weights. This is called "transfer learning", i.e. "transfer" the pre-trained models to the specific problem we are solving.

In transfer learning, these pre-trained weights also show good generalization performance for images outside the dataset.Since the pre-trained model is already well trained, we do not modify the weights too much in a short period of time, and when it is used in transfer learning, we often just fine tune it. Usually we adopt a lower learning rate than the normal training model.

2.1.3.1 Feature extraction

Using pre-trained models as feature extractors. This is done by removing the output layer and then treating the entire remaining network as a fixed feature extractor so that it can be applied to a new data set.

2.1.3.2 Structure using a pre-trained model

Use the structure of a pre-trained model, but first randomize all the weights and then train based on our own dataset.

2.1.3.3 Train specific layers and freeze other layers

Another way to use a pre-trained model is to partially train it. This is done by keeping the weights of some of the layers at the beginning of the model unchanged and retraining the layers that follow to get new weights. In this process, we can try several times to find the best match between frozen layers and retrained layers based on the results.

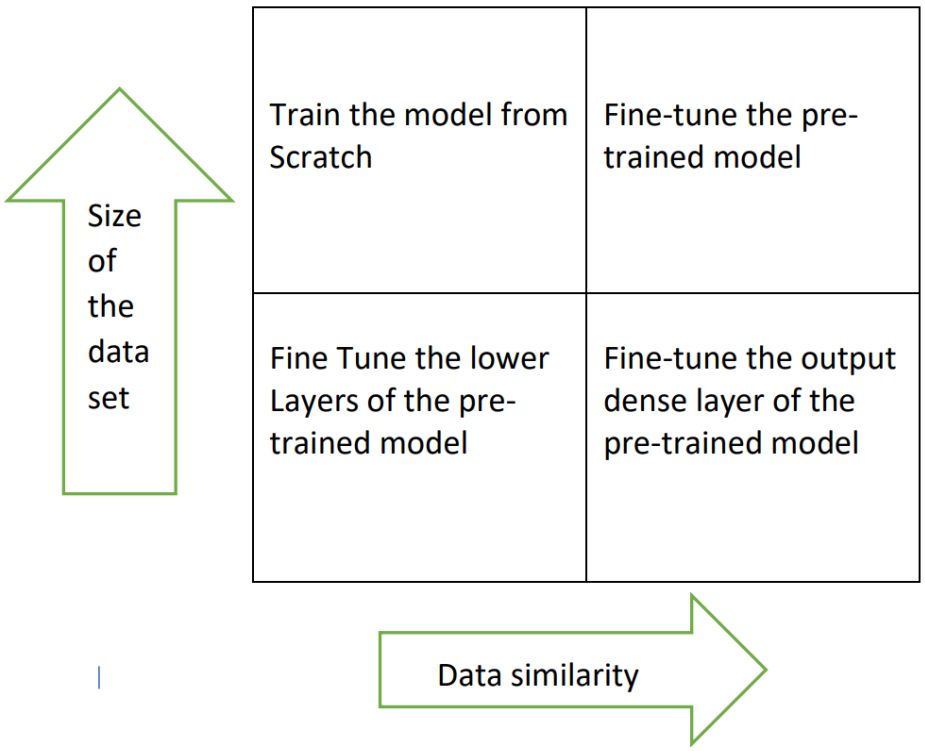

Figure 2.8: Use of transfer learning in different conditions

Figure 2.8 shows four different conditions when use transfer learning according to the size of the dataset and the data similarity.

Scenario 1: Small dataset, high data similarity (compared to training data from pre-trained model): In this case, usually don't need to retrain the model because the data is very similar to the training data of the pre-trained model, just need to change the output layer to match the structure of the problem situation and use the pre-trained model as a feature extractor.

Scenario 2: Small data set, low data similarity: In this case, the usual approach is to freeze the front k-layer, train the output layer, and modify the output layer according to the task.

Scenario 3: Large data sets, low data similarity: In this case, since there are a large dataset, the training process of the neural network will be more efficient. However, because there is a big difference between the real data and the training data from the pre-trained model, using the pre-trained model will not be an efficient way.

Therefore, the best way is to initialize all the weights in the pre-trained model and start the training process from scratch.

Scenario 4: Large data sets, high data similarity: This is the ideal situation where it becomes very efficient to use a pre-trained model. The best way to use it is to keep the original structure and initial weights of the model unchanged and then retrain it on a new data set.

2.2 Related works

Many researchers have made great breakthroughs in computer vision through deep learning techniques

The first issue is feature selection, which is the basic process of acquiring and extracting features used to represent the original data. How to learn the representation of unlabeled data becomes a major consideration. There are many methods to solve this problem.

In [6], Chuen-Kai Shie et al. use Convolutional Neural Network (CNN) to perform transfer representation learning, first learning irrelevant images by unsupervised method, and then modeling relevant data with the learned feature, and finally obtaining 89.63% accuracy. This work shows that transfer presentation learning can solve two challenges of medical image analysis - the lack of labeled data and the lack of domain knowledge. It should be noted that if there are a small number of labeled data in the process of transfer learning, the best performance can be achieved by fine-tuning the whole network with these data [7].From this paper, we know that taking a transfer learning approach can effectively improve the accuracy of the model.

In [8], M. Chen et al. introduce a method firstly utilizes the original image patch to learn unsupervised features and utilizes a small amount of labeled data to fine-tuning. In his paper, Since the image lacks labels, the traditional supervised learning method is not suitable for this situation, so the authors propose to use the auto-encoder method. In addition, the author's training at the patch level also gave us a lot of inspiration for future training.Compared with other data driven methods, the advantages of this method are verified by comprehensive experiments. Moreover, it demonstrates that system performance and feasibility can be affected by data quality because the role of the expert is ignored. Therefore, he concluded that he would combine domain knowledge and data-driven feature learning in his future work.

In [9], Takayasu Moriyaa et al. introduce another method, the main approach consists of two phases. In the first stage, joint unsupervised learning [5] (JULE) is used to learn the deep feature representation of the training patch from the target image. Then, joint unsupervised learning is used to cluster alternately the representation generated by CNN, and cluster labels are used as supervised signals to update CNN parameters. Their main contribution is the combination of JULE

and k-means for medical image segmentation. The label is generated with k-means, and then the labels and patches are combined for parameters update.

Another way to solve the problem of unlabeled data is self-supervised learning, which the author mainly used in this paper. This is a discriminative approach. The main principle is to have the data automatically generate labels. How to let the data generate its own labels? To learn visual features in unlabeled data, a popular solution is to propose various pretext tasks to be solved for the network and learn features by learning these pretext tasks. Some different pretext tasks have been proposed including colorizing grayscale image inpainting [10], image colorization [11], image jigsaw puzzle[12] etc.

Self-supervised learning has also been well explored in many areas, in [13], Mehdi Noroozi et al. hope to conduct self-supervised learning through the information of spatial equal variance, that is, the feature summation obtained after a picture is divided into several parts should be equal to the direct feature summation of the whole picture. In practice, if there are no other constraints, such learning method will surely lead to a trivial solution, that is, all outputs are 0. In order to avoid such problems, the author also added another negative sample into the objective function as a constraint, that is, the feature summation of these parts is required to be as different as possible from the feature representation of an unrelated image of any sample.

self-supervised learning has also been explored in medical imaging domain, in [14],this paper applies a self-supervised learning approach to cardiac image segmentation by predicting anatomical locations.The anatomical position serves as a supervisory signal and does not require additional manual annotation. Most previous studies on cardiac MR image segmentation consider the short axis and long axis images separately, ignoring the relative directions of different images. However, in this paper, the author proposed that using the relative directions of short-axis view and long-axis view as well as the anatomical position defined by the view plane, a pretext task could be developed to train the network in a self-supervised way and improve the data efficiency.In addition, the author used three different transfer learning methods and obtained good performance through comparison.

in [1] Zhongwei Zhou et al.by setting up a mechanism for self-supervised learning, which can generalize well to multiple tasks. They concluded that self-supervised learning is not about data collection and professional labeling, but about designing an effective self-learning mechanism. The proposed approach to self-supervised learning boils down to making some changes to the original images, and then letting the model restore the original images. In this way, the original image itself becomes a label for supervised model training, in line with the original intent of self-supervised learning, learning features directly from the data. They set up four pretext tasks for self-supervised learning: non-linear transformation,local-shuffling,

in painting,out painting.These four different image transformations can be easily combined to give a given image the ability to undergo one or more transformations at the same time. Eventually they are all absorbed into one "image recovery" task, giving the model the ability to learn visual features.

We decided to use 'model gensis' as the starting point for our thesis because we found that model genesis yielded results that were ahead of all other methods of self-supervised learning approach. Moreover, in this paper, the authors train a separate model for each modality, so we hypothesize that we can train all the modalities together to achieve real 'model genesis'.

Chapter 3

Materials and methods

In this chapter, we will provide a detailed description of the dataset used in this thesis, including an introduction to the dataset, dataset preprocessing, dataset splitting, and image sampling strategies.

3.1 Materials

3.1.1 The introduction of datasets

The data comes from 21 different datasets containing four different modalities:

1. Radiography (X-ray)
2. Computed tomography (CT)
3. Magnetic Resonance Imaging (MRI)
4. Mammogram

The images in each dataset include information on different diagnosis, different body parts, different dimensions, etc. Usually CT and MRI images are always stored in 3D, while X-ray images are usually stored in 2D. In this thesis our pre-training model is done using 2D images. To maintain the diversity of the data, this study tried to ensure that the dataset included as many different patients, different body parts, and different diagnosis as possible, and based on this, I recreated csv files in a uniform format for this information.

I re-partitioned these datasets into two parts, 17 of which were used for the pre-training of the model, which is mainly related to self-supervised learning, and the rest of the dataset will be the training set, which will be used for the transfer learning of the model. I divided the pre-training datasets into training, testing,

and validation sets based on different patients, which will be described in more detail in the following sections.

3.1.2 Datasets

In this section, the use of these datasets are described in more detail, and those datasets were divided into two parts, one for the pre-training set for self-supervised learning and the other part of the training set for transfer learning.

3.1.2.1 Pre-training datasets

1. **CT Lymph Nodes:** This collection consists of Computed Tomography (CT) images of the mediastinum and abdomen in which lymph node positions are marked by radiologists at the National Institutes of Health, Clinical Center. Radiologists at the Imaging Biomarkers and Computer-Aided Diagnosis Laboratory labeled a total of 388 mediastinal lymph nodes in CT images of 90 patients and a total of 595 abdominal lymph nodes in 86 patients.[15]

2. **Medical Segmentation Decathlon[16]:**

   (a) **Liver:** The modality of the dataset is CT, which contains 201 3D volumes (131 Training + 70 Testing)

   (b) **Prostate:** The modality of the dataset is Multimodal MR (T2, ADC), which contains 48 4D volumes (32 Training + 16 Testing), in end use. We removed the ADC.

   (c) **Colon:** The modality of the dataset is CT, which contains 190 3D volumes (126 Training + 64 Testing)

   (d) **Spleen:** The modality of the dataset is CT, which contains 61 3D volumes (41 Training + 20 Testing)

   (e) **Hepatic Vessel:** The modality of the dataset is CT, which contains 443 3D volumes (303 Training + 140 Testing)

   (f) **Pancreas:** The modality of the dataset is CT, which contains 420 3D volumes (282 Training +139 Testing)

3. **CQ500 Dataset:** They made the CQ500 dataset of 491 scans with 193,317 slices publicly available so that others can compare and build upon the results they has achieved in the paper. and provide anonymized dicoms for all the 491 scans and the corresponding radiologists' reads[17].

4. **CHAOS:** Two databases are used in the challenge: Abdominal CT and MRI (T1 and T2 weighted). Each data set in these two databases corresponds to

a series of DICOM images belonging to a single patient. The data sets are collected retrospectively and randomly from the PACS of DEU Hospital. There is no connection between the data sets obtained from CT and MR databases (i.e. they are acquired from different patients and not registered)[18].

5. **Deep Lesion:** The National Institutes of Health's Clinical Center has made a largescale dataset of CT images publicly available to help the scientific community improve detection accuracy of lesions. While most publicly available medical image datasets have less than a thousand lesions, this dataset, named DeepLesion, has over 32,000 annotated lesions identified on CT images.The images, which have been thoroughly anonymized, represent 4,400 unique patients.[19].

6. **Mrnet:** The MRNet dataset consists of 1,370 knee MRI exams performed at Stanford University Medical Center. The dataset contains 1,104 (80.6%) abnormal exams, with 319 (23.3%) acl tears and 508 (37.1%) meniscal tears; labels were obtained through manual extraction from clinical reports[20].

7. **MURA:** MURA (musculoskeletal radiographs) is a large dataset of bone X-rays. Algorithms are tasked with determining whether an X-ray study is normal or abnormal[21].

8. **Cardiac MRI:** Cardiac MR images acquired from 33 subjects.Each subject's sequence consists of 20 frames and 8-15 slices along the long axis, for a total of 7980 images. The sequence corresponding to each subject x is in a distinct .mat (MATLAB) file named sol_yxzt_patx.mat. These are the raw, unprocessed images, that were originally stored as 16-bit DICOM images[22].

9. **NIH Chest X-ray Dataset of 14 Common Thorax Disease Categories:** The modality is X-ray, body part is chest, and contains 30,805 unique patients,112,120 X-ray images.% [23].

10. **Brain Tumor dataset:** This brain tumor dataset contains 3064 T1-weighted contrast-inhanced images with three kinds of brain tumor. Detailed information of the dataset can be found in readme file [24].

11. **OASIS:**This set consists of a cross-sectional collection of 416 subjects aged 18 to 96. For each subject, 3 or 4 individual T1-weighted MRI scans obtained in single scan sessions are included. The subjects are all right-handed and include both men and women. 100 of the included subjects over the age of 60 have been clinically diagnosed with very mild to moderate Alzheimer's disease (AD). Additionally, a reliability data set is included containing 20 nondemented subjects imaged on a subsequent visit within 90 days of their initial session[25].

12. **IBSR:** Eighteen subjects are currently available.For each subject there is: T1-weighted volumetric images that have been 'positionally normalized' into the Talairach orientation (rotation only). Also note that these data have been processed by the CMA 'autoseg' biasfield correction routines; unbias-field-corrected is also available. Segmentation results for 'general segmentation' of 43 individually labeled principle gray and white matter structures of the brain[26].

3.1.2.2 Training datasets

The two datasets used for transfer learning are described in detail below:

1. **Luna16:** The modality is CT scan, the body part is lung, and the dataset includes 888 patients, 888 images, 551065 patches, 1186 lesions, and 36378 annotations class label (0 for non-nodule and 1 for nodule)can be multiple candidates per nodule.

2. **MURA:** MURA (musculoskeletal radiographs) is a large dataset of bone X-rays. Algorithms are tasked with determining whether an X-ray study is normal or abnormal[21].

3.1.3 Data sampling and split

3.1.3.1 Data sampling

Fig.3.1 is a summary of these 17 datasets, with a total number of images CT (452623), X-RAY (148928), and MRI (429290) in the original data set. From the quantity, it can be seen that the number of X-RAY is too small relative to the number of images of the other two modalities, because we decided to train the images of the three modalities, the sample number unbalance may lead to some problems, for this situation, data sampling is the best solution to address data unbalance.

dataset	modality	#slices	body part	#patient	mean #images per patient	max value	min value
CHEST X-RAY14	X-RAY	112120	chest	30805	3.6	184	1
MURA	X-RAY	36808	WR_WRIST(9752) XR_SHOULDER(8379) XR_HAND(5543) XR_FINGER(5106) XR_ELBOW(4931) XR_FOREARM(1825) XR_HUMERUS(1272)	11184	3.3	29	1
CQ500	CT	169037	BRAIN	488	346.4	1949	4
CT LYMPH NODES	CT	110003	ABDOMEN(54512) MEDIASTINUM(54512)	176	625	776	56
LIVER	CT	85679	LIVER	201	426	1026	42
PANCREAS	CT	26719	PANCREAS	282	94.7	751	28
DEEP LESION	CT	22919	INFO NOT FOUND	3060	7.4	114	1
HEPATICVESSEL	CT	21120	HEPATICVESSEL	303	69.7	181	24
COLON	CT	13486	COLON	126	107	729	37
SPLEEN	CT	3650	SPLEEN	41	89	168	31
Oasis1	MRI	216000	BRAIN	428	128	128	128
IBSR	MRI	12522	BRAIN	20	128	128	128
BrainTumour	MRI	3056	BRAIN	42	20	32	15
MRNET	MRI	118109	KNEE	1456	81	162	19
CARDIAC MRI	MRI	1980	CARDIAC	33	241	300	160
CHAOS	MRI	1917	ABDOMEN	20	96	136	78
PROSTATE	MRI	1204	PROSTATE	32	37	48	22

Figure 3.1: summary of the pre-training datasets

As can be seen from Fig.3.1, there are a total of 19 different body parts in 17 different datasets, plus an info not found (deep lesion), deeplesion dataset contains images from different body parts, but labels are not available.

3.1.4 Explanation of the datasets

During sample, a patient often contains many 2D images(X-ray) or 3D images(CT and MRI), because we consider training from 2D images, we can split a 3D image into different 2D images. However, the number of images contained in a patient varies, and the images are usually down-sampled for each patient, e.g., if a patient has 600 2D images, 300 of them are taken so that no information on the total number of slices is missed. Also, during the sampling process, only sampled the number of images that the patient has and do not sample the number of patients. In summary, the images of each patient were sampled while maintaining the number of patients in the data.

The following can be obtained from Fig.3.1.

1. **the number of patients:** Since the sample is based on patient level, the

number of patients information is the first point of concern.

2. **mean number of images per patient:** The number of slices per patient.
3. **max value:** The number of images that the patient has with the highest number in the dataset.
4. **min value:** he number of images that the patient has with the lowest number in the dataset.

From those parameters it's easily concluded that the mean of the number of slices that the patient has, and the difference between the maximum, and thus judge whether the mean value can be used as the number of samples. If the difference is too small, sample with the mean value(for example, In dataset MRNET), if the difference is too large, then cannot sample with the mean(for example in deep lesion).

Through these, the distribution of the number of slices owned by the patients in this data set were ealisy to be understood. so that not lose too much information by oversampling.

3.1.5 Image sampling strategy

There are about a million images in the source dataset in total, the goal is to sample as many images as possible while avoiding too strong unbalance and removing images that are strongly correlated with each other.

Since the number of X-RAY in the source dataset is only 140k, the number of images is small relative to the other two modalities, and for the purpose of balancing the dataset, we tried to sample images of the other two modalities while controlling the number of X-RAY.

Another problem is that the body parts of the dataset are very unbalance distributed, with brain being far too numerous, taking up more than 46% of the dataset, and prostate being too small.

In the process of analyzing the data, we have separately considered the number of patients, body parts, modalities, diagnosis, etc. In order to maintain the diversity of the data, and after the pre-analytical experiments, we have come to the conclusion that it is not possible that sample based on diagnosis because most of the datasets lack information about diagnosis, which results in part of the dataset not being collected during the sampling process.

we are sampling mainly for modalities, body part, and patients. the sampling is all based on patient, maintaining the balance of modalities, and body part diversity and balance. For different datasets, this study decided to sample in different ways, and all were based on patient. The following is an explanation of the sample method for each dataset:

1. **MRNET:** Since more than half of the data during the sample of BRAIN were removed, resulting in a not particularly large number of MRI, in this thesis we selected a larger number of images to maintain balance when sampling MRNET, and from Fig. 3.1, the difference between the mean value and max values,so selecting the mean value as the sample size for each patient was a relatively good choice

   In the end, I decided to select all the patients (1250 patients), taking 81 slices per patient

2. **COLON:** The same strategy as for MRNET, since the differences between these values are not large, the number that around mean value is chosen as the number of samples. In the end, I decided to select all patients (126 patients), 80 slices per patient.

3. **CARDIAC MRI:** Since the mean value is 241, but the maximum is only 300, so instead of sampling based on the mean, and since the number of MRI is not enough, I chose 200 images per patient. In the end, I decided to select 33 patients, 200 slices per patient.

4. **CHEST X-RAY14:** Because the number of X-RAY is small in the source dataset, so a large number of images has to be sampled, where most patients had only one or two slices and the mean was only 3.6, but the maximum was 184, since more than 20% of patients in the dataset have more than 40 images, it cannot keep only 2-3 per patient. For these reasons, I decided to choose 20 as the sample value. In the end, I decided to select all patients (30805) with a sample value of 20, below which the author collected the number of patients that each patient currently has.

5. **DEEP LESION:** I decided to keep all the images in this dataset. The reason is that the dataset was already downsampled by the authors of DEEP LESION , since it is enriched with lesions. It is also important to keep this dataset since it contains a lot of tumors, and hence have to expose the network to both normal and abnormal anatomy.

6. **LIVER:** Same as what I did in MRNET, those value are close so that the mean value can be token as the sample size. I decided to keep all patients (201 patients) with 100 slices per patient.

7. **PANCREAS:** Same as Liver I decided to keep all patients (281 patients) with 80 slices per patient.

8. **CT LYMPH NODES:** Same as liver I decided to keep all patients (176 patients) with 100 slices per patient.

9. **HEPATICVESSEL:** Same as PANCREAS I decided to keep patients (303 patients) with 69 slices per patient.

10. **CQ500 DATASET:** Because this dataset contains images of the brain, but modality is CT, unlike MRI in BrainTumour, but it is the same body part, so this thesis sampled the same amount as previously to balance the number in BrainTumour. I decided to choose all patients (488 patients), 100 slices per patient.

11. **CHAOS:** The dataset was too small. I decided to sample it all.

12. **SPLEEN:** The dataset was too small. I decided to sample it all.

13. **PROSTATE:** The dataset was too small. I decided to sample it all.

14. **BrainTumour:** The dataset was too small. I decided to sample it all.

15. **IBSR:** The dataset was too small. I decided to sample it all.

16. **MURA:** Although the dataset is large, it contains five different body parts, and the number of x-rays in the source dataset is small, for these reasons, all are sampled.and after the above strategy, I ended up with a selection of 450K images.

17. **Oasis:** The Oasis datasets contains 3 sub datasets, Oasis1,2,3, and I sample all patients, each patient with 100 slices.

3.1.6 Analysis of data sampling results

With the analysis in the previous section, we took different strategies, and the dataset was eventually balanced.

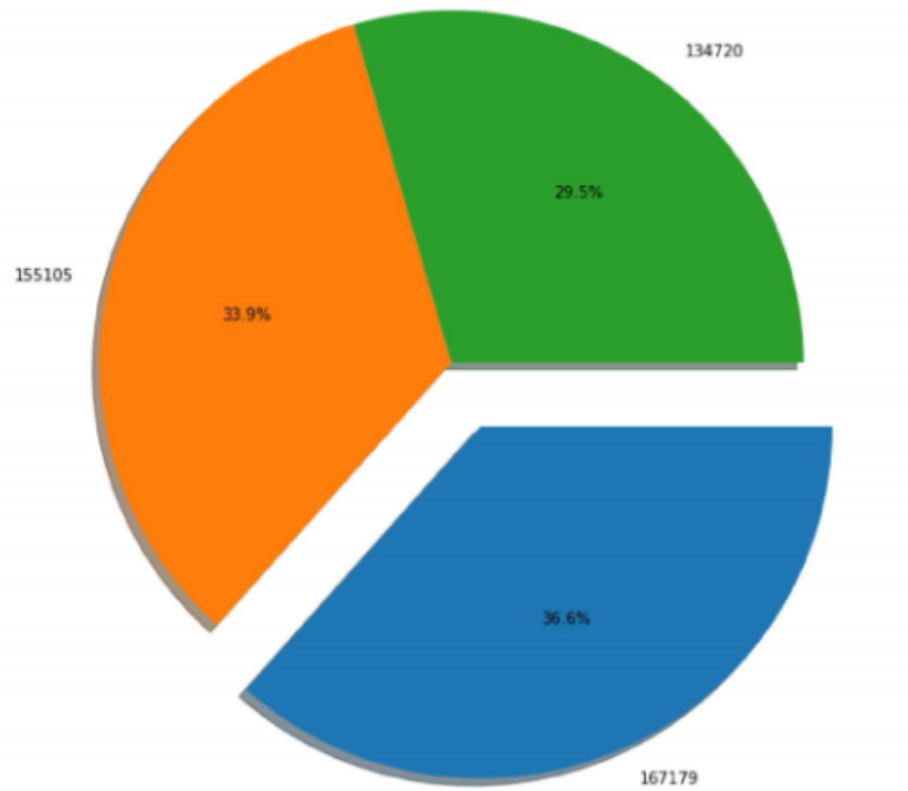

Figure 3.2: The number of images in each modality

Finally, the number of images in MRI is 167179, the number of images in CT is 155105, and the number of the images in X-RAY is 134720.

Fig.3.2 shows that the number of images in each modality,it can be seen that in the end modality is balanced.

BODY PART	
CHEST	97912
KNEE	96554
BRAIN	90591
INFORMATION NOT AVAILABLE	22919
PANCREAS	21378
LIVER	19764
HEPATICVESSEL	15516
ABDOMEN	10517
XR_WRIST	9752
COLON	9531
MEDIASTINUM	8956
XR_SHOULDER	8379
CARDIAC	6340
XR_HAND	5543
XR_FINGER	5106
XR_ELBOW	4931
SPLEEN	3650
XR_FOREARM	1825
XR_HUMERUS	1272
PROSTATE	1204

Figure 3.3: The number of slices in each modality

As can be seen from Fig.3.3 and Fig.3.4, although the body part is not fully balanced, the distribution has improved considerably relative to the source dataset. The body part in the number of brains has dropped from 46% to 20% compare with the original dataset.

Fig.3.5 shows that the number of slices in each dataset.

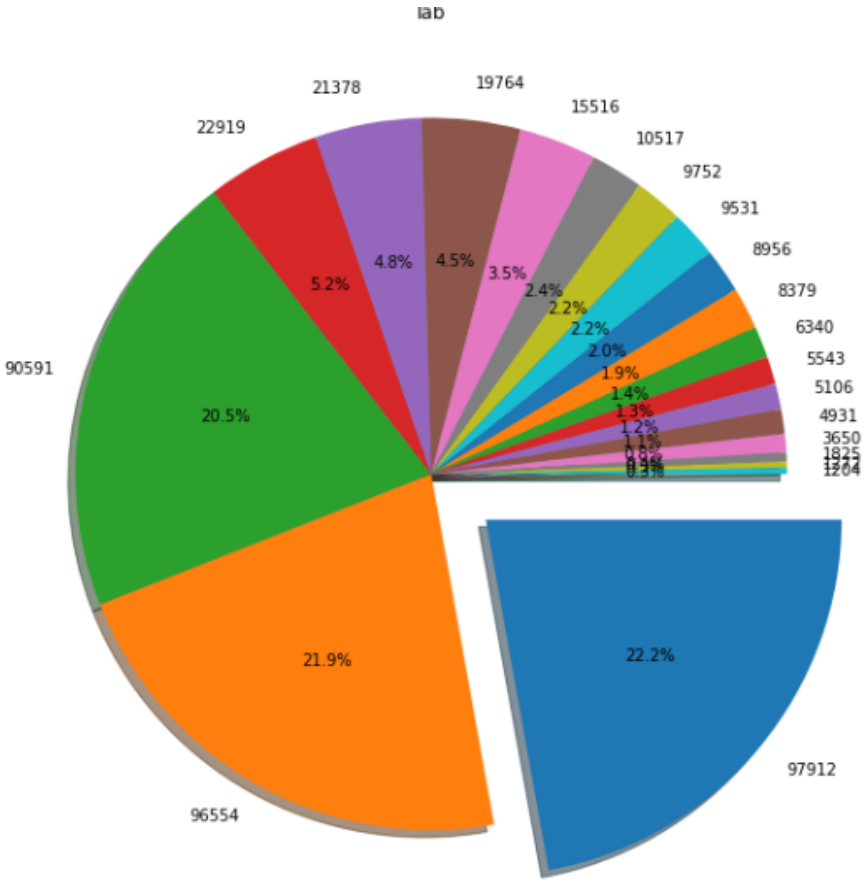

Figure 3.4: The percentage of slices in each dataset

DATASET	
CHEST X-RAY14	97912
MRNET	96554
BrainTumour	45800
CQ500 DATASET	44791
MURA	36808
DEEP LESION	22919
PANCREAS	21378
LIVER	19764
CT LYMPH NODES	17556
HEPATICVESSEL	15516
COLON	9531
CARDIAC MRI	6340
SPLEEN	3650
CHAOS	1917
PROSTATE	1204

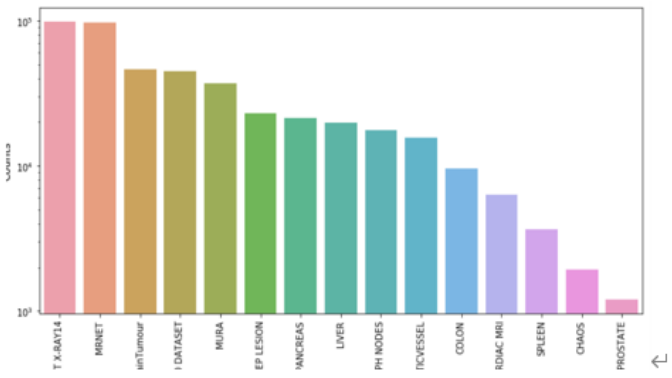

Figure 3.5: The number of slices in each dataset

Fig.3.6 shows that the number of patients in each dataset.

GROUP BY NAME DATASET AND PATIENT

Name Dataset	
BrainTumour	459
CARDIAC MRI	32
CHAOS	20
CHEST X-RAY14	30804
COLON	126
CQ500 DATASET	488
CT LYMPH NODES	176
DEEP LESION	3059
HEPATICVESSEL	304
LIVER	201
MRNET	1250
MURA	18515
PANCREAS	281
PROSTATE	32
SPLEEN	41

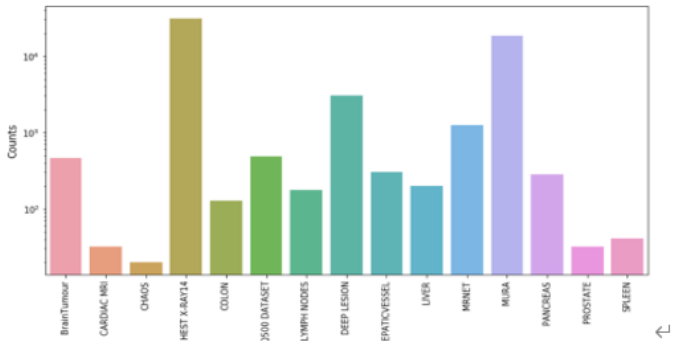

Figure 3.6: The number of patients in each dataset

Fig.3.7 shows that the body part in each modality.

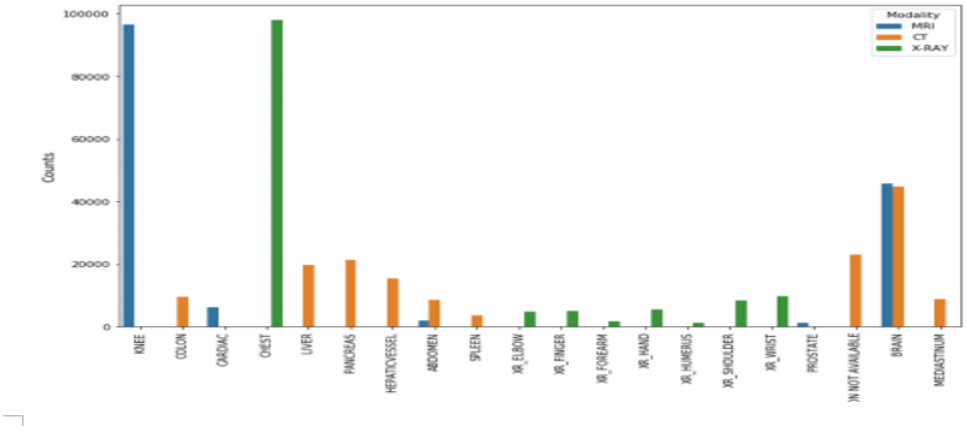

Figure 3.7: the body part in each modality

Finally, there are 1793 patients with modality MRI, 49319 patients with modality X-RAY, and 4676 patients with modality CT.

3.1.7 Data split

The data set was split according to the patient's level to ensure that the patients appearing in the test set and the validation set did not appear in the training set.

For this reason, I decided to select 90% of the images as the training set, 5% of the images as the test set and 5% of the images as the validation set.

3.1.8 Pre-processing the images

This section focus on pre-processing for three different modalities, and the goal is to uniformly plan the pixel value to between 0 and 1 for images with large differences, in order to facilitate convergence for model training at a later stage.

The main technique we used is normalization, and the pre-processing is done dynamically for each 2D image for the different modalities images.

3.1.8.1 Pre-processing CT

There are 8 datasets in the dataset that are CT, containing different data formats, DICOM, nii, png, mhd, and for different data formats, different loading methods has been taken. The goal is to normalize the pixel value of the input image (-4000 4000) to [0,1] interval.

This procedure is done using the following formula(Fig.3.8):

$$image = \frac{image - MIN_{BOUND}}{MAX_{BOUND} - MIN_{BOUND}}$$

Figure 3.8

I set the MIN_BOUND to -1000 and MAX_BOUND to 1400.

The current value range is [-1024,2000]. And values arbitrarily greater than 1400 do not need to be considered, since they are all artifacts.

The processing is the same for all other CT datasets, the only difference is that for the deeplesion dataset, the image format is png, we first converted the .png image to .nii using the officially provided processing script, the other processing method for CT in all the same only the loading to read the image is different.

Finally, I decided to perform random clip for images of size 448*448, and resize these random clippings in the proposed size as input, i.e. 224*224.

And I plotted out the image changes before and after the pre-processing, the pixel distribution as follows:

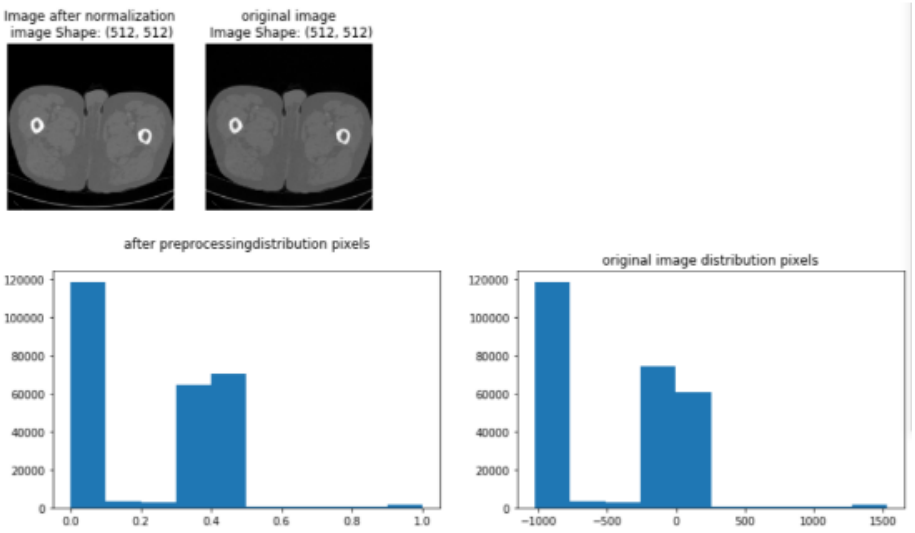

Figure 3.9: comparison before and after preprocessing for CT Lymph node dataset

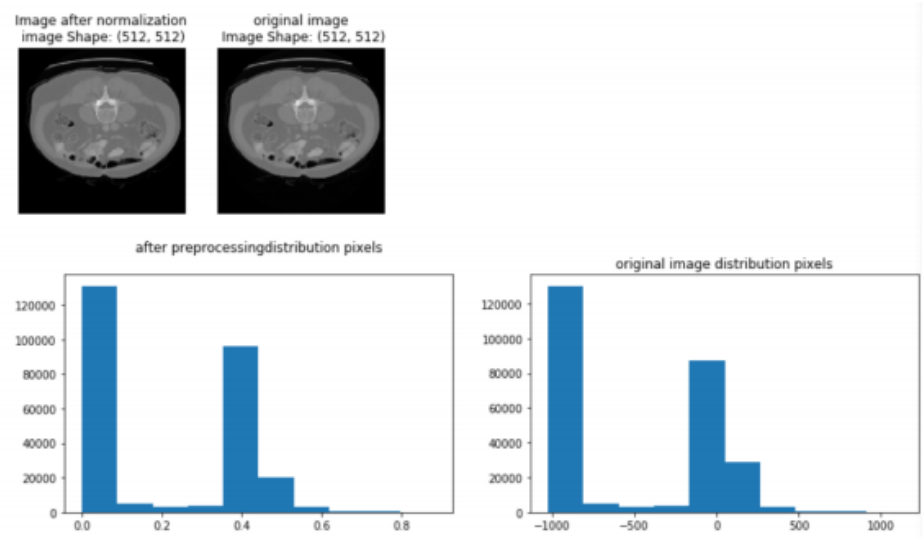

Figure 3.10: comparison before and after preprocessing for spleen dataset

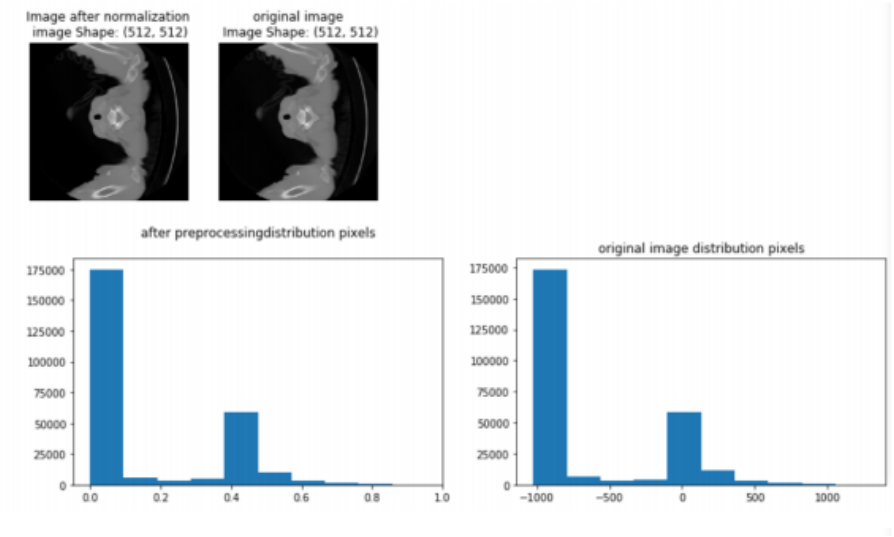

Figure 3.11: comparison before and after preprocessing for deeplesion dataset

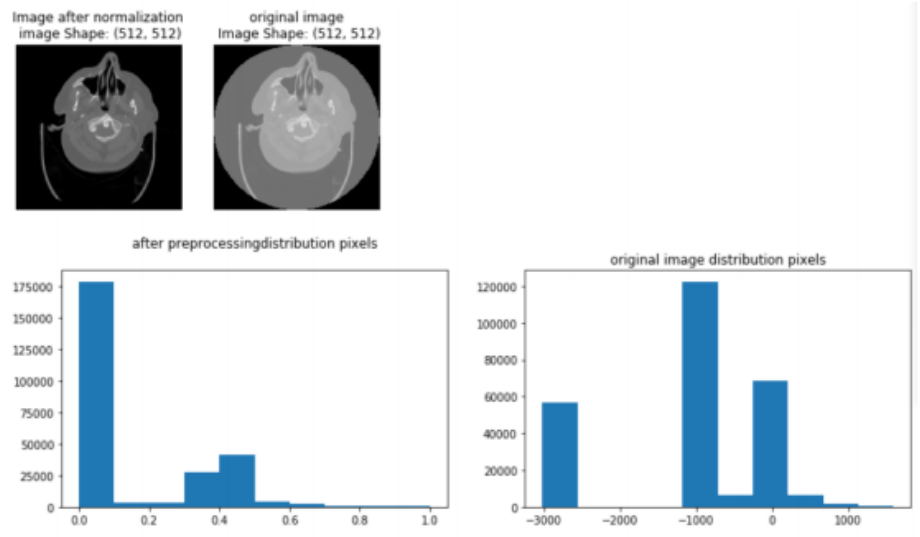

Figure 3.12: comparison before and after preprocessing for CQ500 dataset

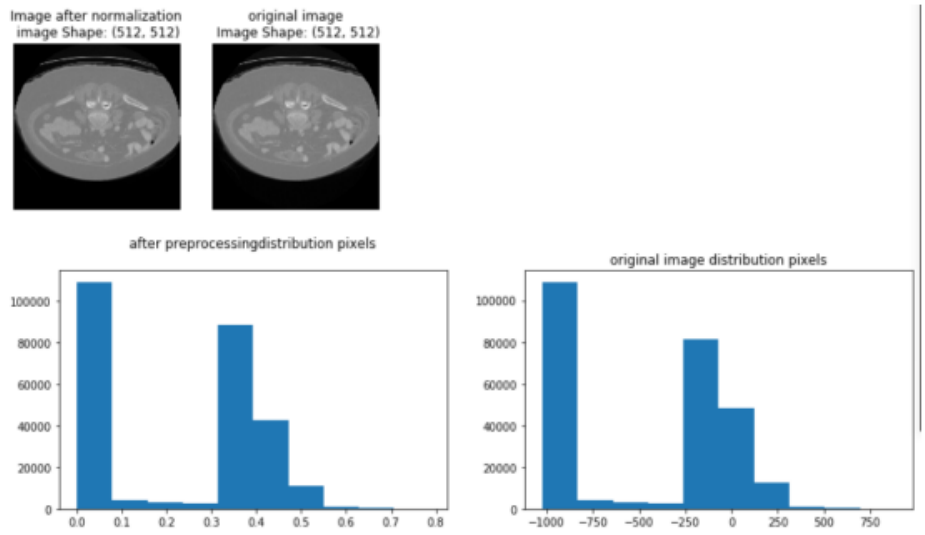

Figure 3.13: comparison before and after preprocessing for HepaticVessel dataset

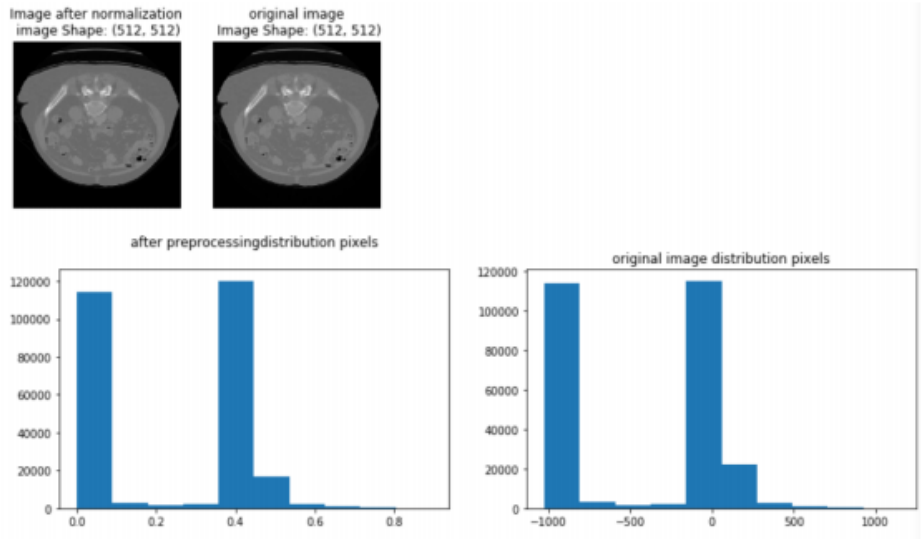

Figure 3.14: comparison before and after preprocessing for Pancreas dataset

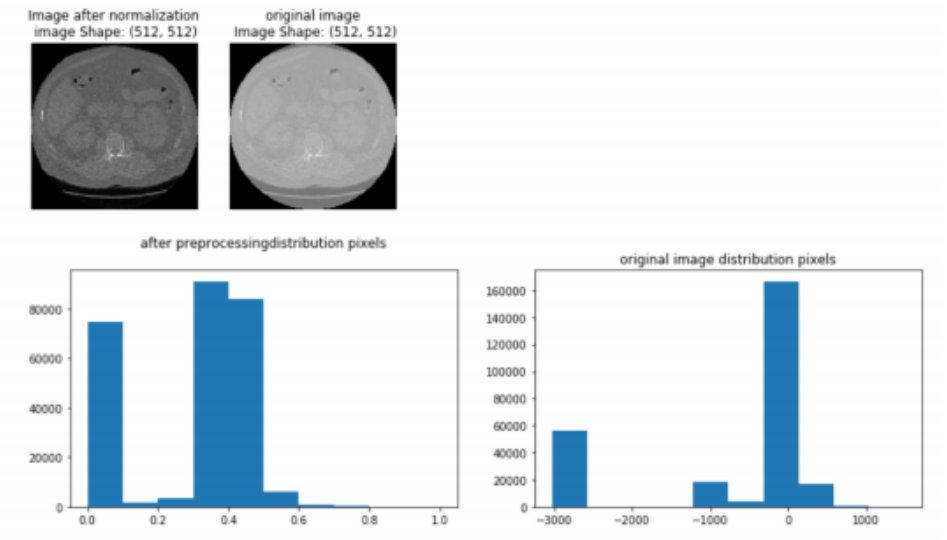

Figure 3.15: comparison before and after preprocessing for Luna16 dataset

3.1.9 Pre-processing MRI

Eight of those datasets are MRI and contain a variety of different image formats, as the models are in 2D image basis for training, so the pre-processing approach I took was also based on 2D images, and the goal is also normalize to the 0-1 range, taking the approach of standard normalization between 0 and 1, I first calculate max pixel value for each 2D image and then divides by this maximum value, thus pre-processing the image to the 0-1 interval, using the advantage of this method is that the image can be dynamically pre-processed.

Finally, I performed random clip for images of size 448*448, and resize these random clippings in the proposed size as input, i.e. 224*224.

And I plotted out the image changes before and after the pre-processing, the pixel distribution as follows:

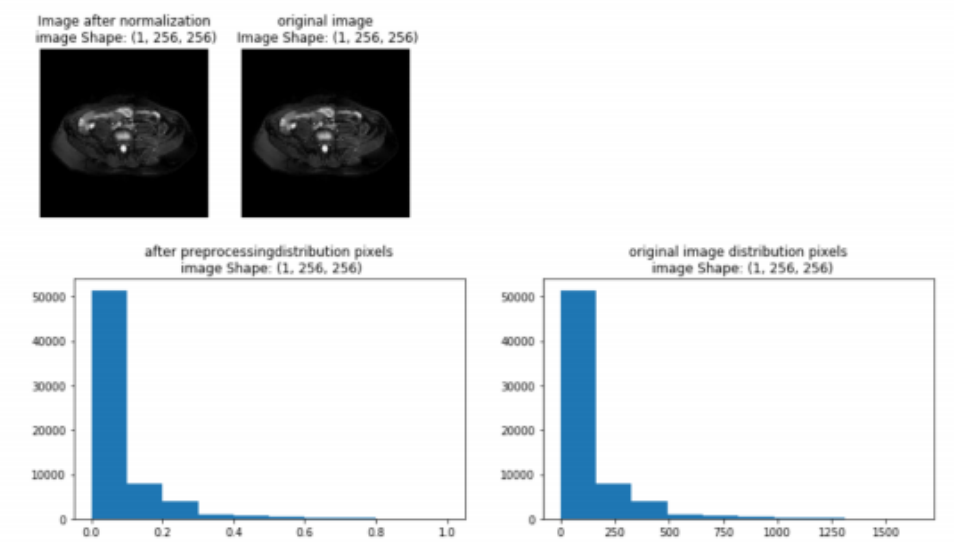

Figure 3.16: comparison before and after preprocessing for chaos dataset

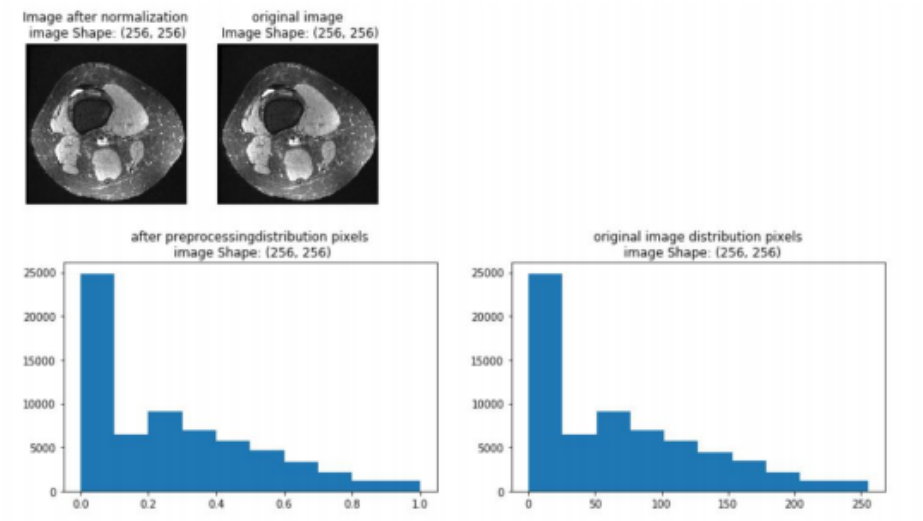

Figure 3.17: comparison before and after preprocessing for Mrnet dataset

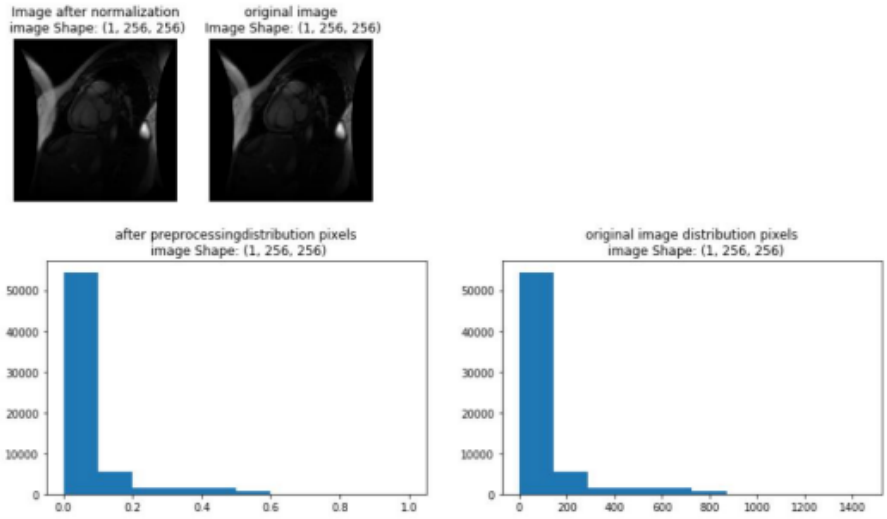

Figure 3.18: comparison before and after preprocessing for Cardiacmri dataset

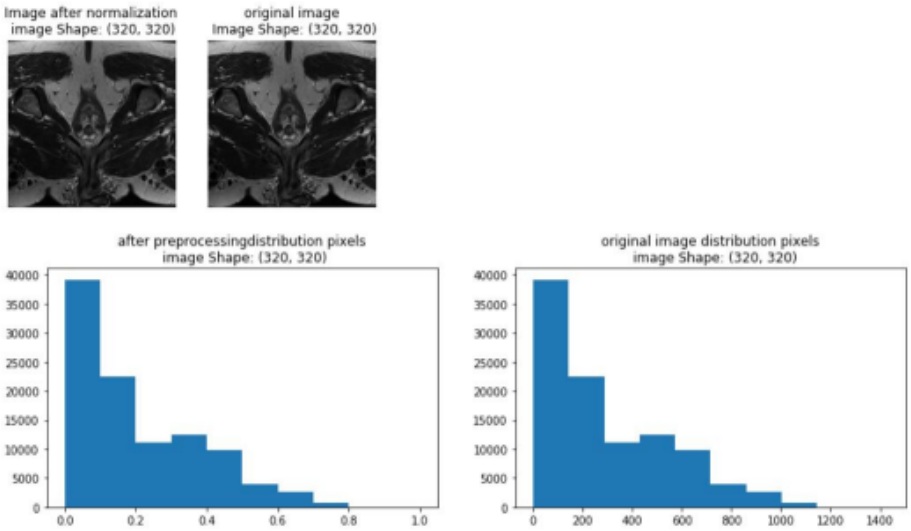

Figure 3.19: comparison before and after preprocessing for Prostate dataset

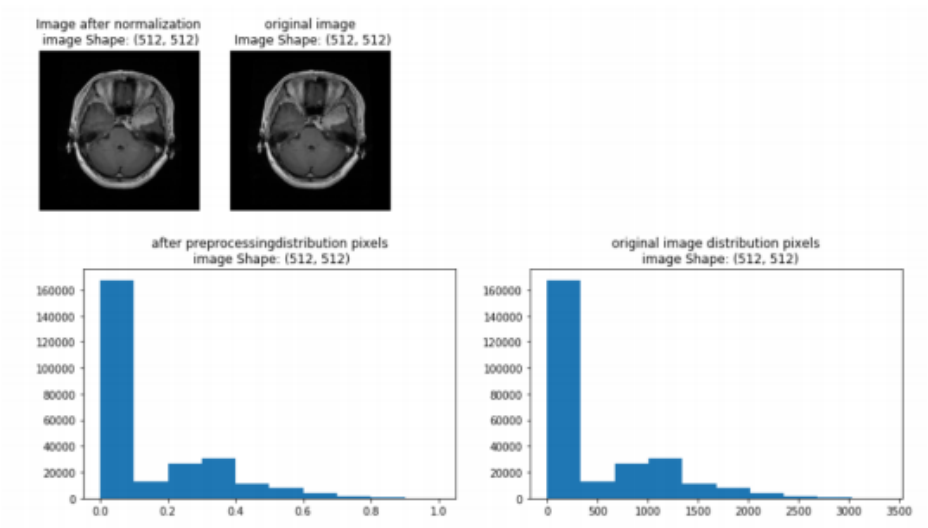

Figure 3.20: comparison before and after preprocessing for BrainTumor dataset

Figure 3.3.3.14 preprocessing IBSR

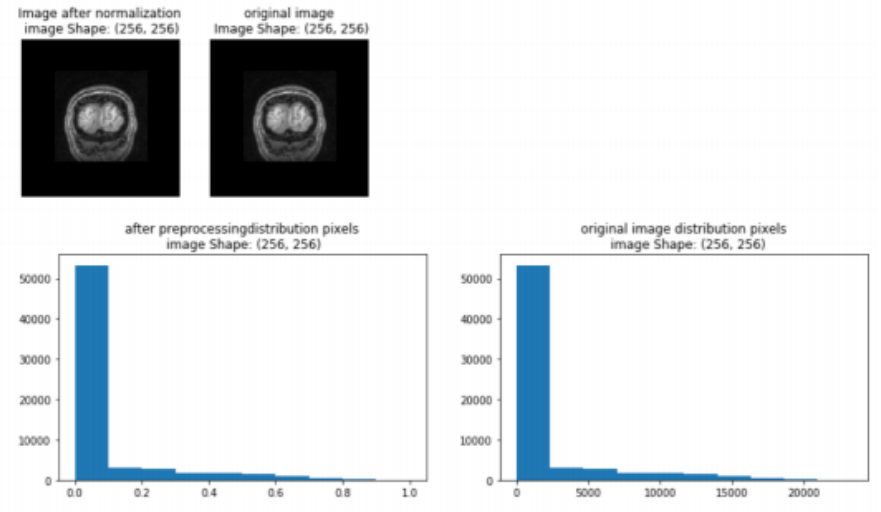

Figure 3.21: comparison before and after preprocessing for IBSR dataset

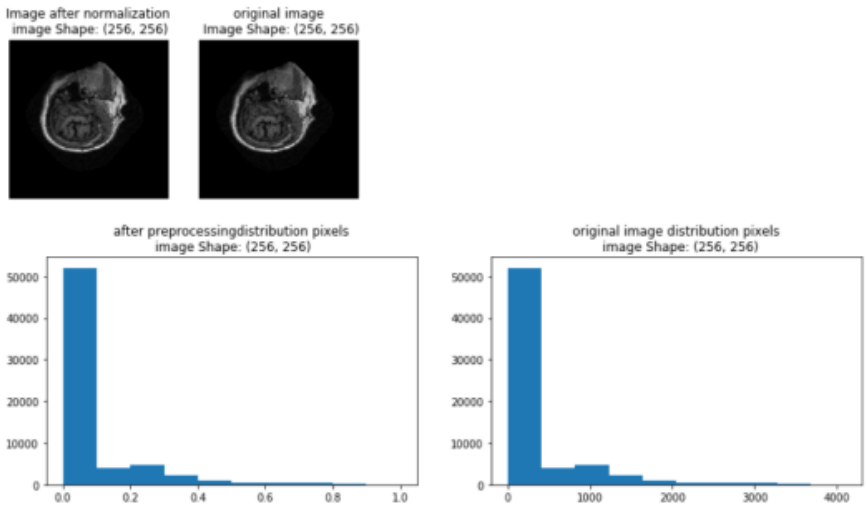

Figure 3.22: comparison before and after preprocessing for OASIS dataset

3.1.10 Pre-processing X-RAY

The pixel values are integers with values between 0 and 255. it is good practice to normalize the pixel values so that each pixel value has a value between 0 and 1. This can be achieved by dividing all pixel values by the 255. This runs on all channels, regardless of from the actual range of pixel values in the image. I convert the image to ndarray float32 and divide ndarray by 255.0. In this way we have normalized the image pixel distribution in the [0-1] range.

Finally, I performed random clip for images of size 448*448, and resized these random clippings in the proposed size as input, i.e. 224*224.

The only different is in MURA, also need a zero padding because of the image size, because some images do not exceed 224 pixels in length or width.

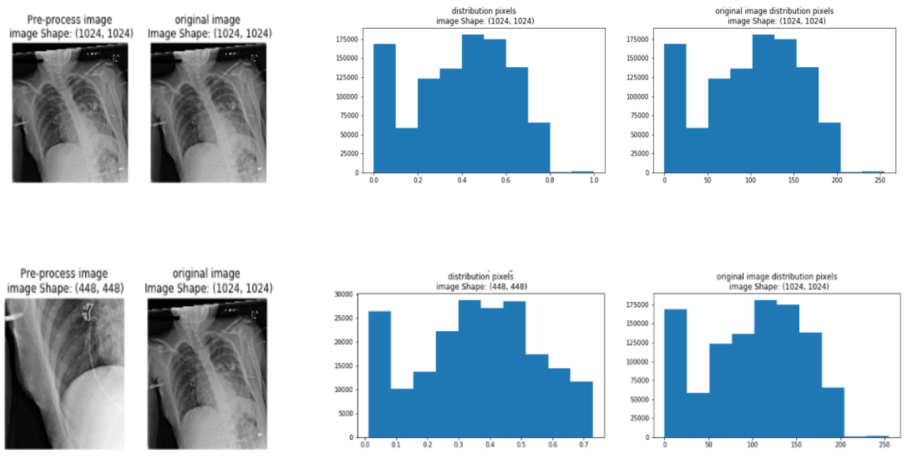

Figure 3.23: comparison before and after preprocessing for chest x-ray14 dataset

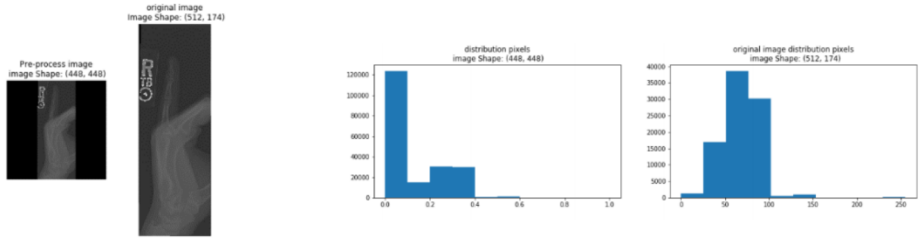

Figure 3.24: comparison before and after preprocessing for Mura dataset

3.1.11 Batch Normalization VS Instance Normalization

For the difference between batch normalization and instance normalization, see the following figure.

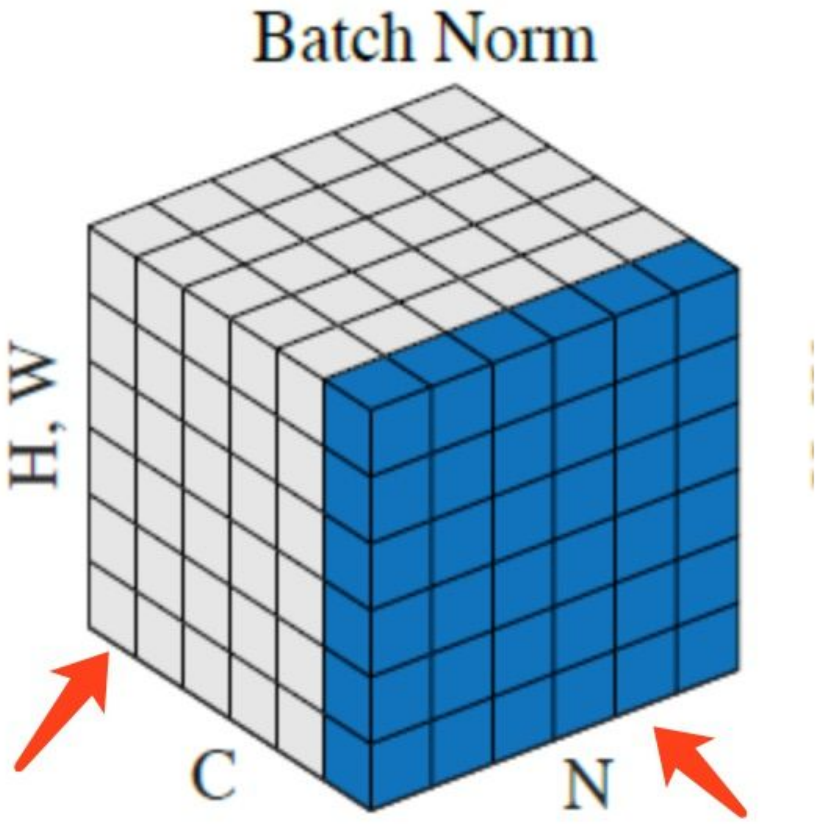

Figure 3.25: An example of Batch Normalization [27]

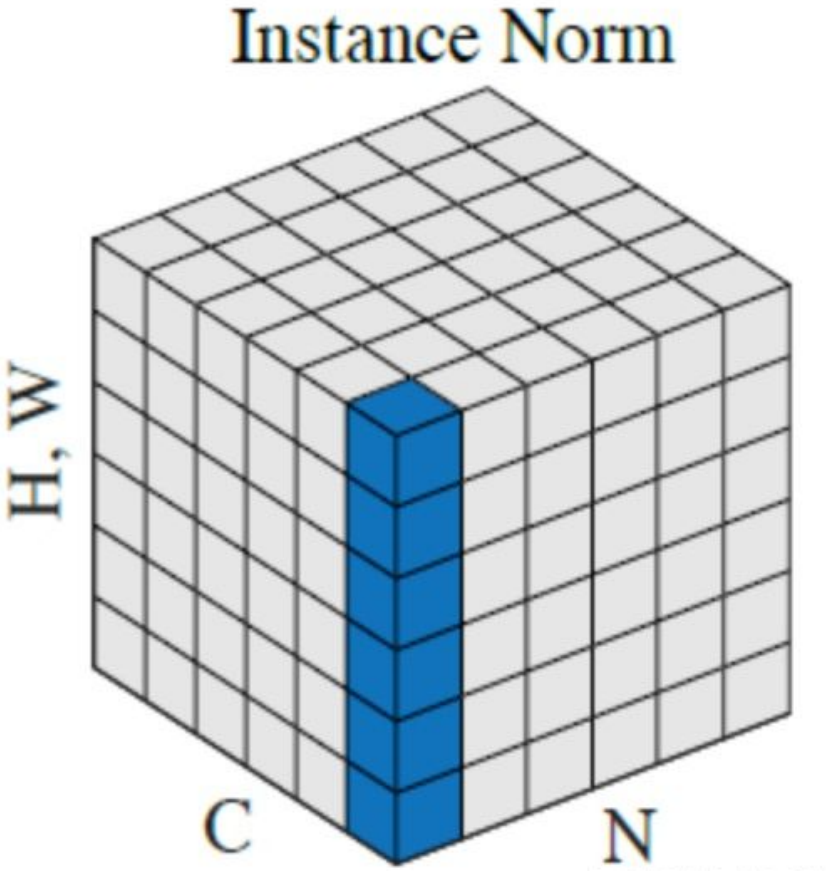

Figure 3.26: An example of Instance Normalization[27]

From this, it is clear that batch normalization means that the same channel of each of the 6 images is normalized together. Whereas instance normalization is the normalization operation performed separately for a single channel of a single image.

Batch normalization is applicable in discriminative models, such as the image classification model. Because batch normalization focuses on the normalization of each batch to ensure the consistency of the data distribution, the results of the discriminant model depend on the overall distribution of the data. However, batch normalization is sensitive to the size of the batch size, because the mean and variance are calculated on one batch each time, so if the batch size is too small, the mean and variance are not enough to represent the whole data distribution; instance normalization is suitable for generative models, such as image style transfer. Since the result of image generation mainly depends on an image instance, normalizing the whole batch is not suitable for image styling, and using instance normalization

in style transfer can not only speed up model convergence, but also maintain the independence between each image instance.

In [2], it has been demonstrated in the segmentation task of prostate, pancreas, and lung nodules. Instance normalization is better than batch normalization, and this thesis also implements instance normalization.

Chapter 4

Experiments and results

4.1 Experiments

Medical image analysis and natural image analysis are not much different in terms of methods, but both use the most popular deep neural networks to do classification, segmentation, detection, etc. Compared with natural image analysis, medical image analysis is very difficult to label, and it is often difficult to match the size of natural image database. When encountering such a small amount of annotation data, "transfer learning" is a popular method, because the model is not learned from scratch, but from a large number of pre-trained models in the data set.

4.1.1 Pre-trained model

In fact, in medical imaging processing, transfer learning from ImageNet pre-trained models has become a standard, and the programming implementation is very simple.

But there is a problem in transfer pre-trained models from ImageNet to medical imaging. Medical images are so different from natural images, however, they are based on a completely different imaging principle than natural images. Therefore, the effect of transfer learning from natural images to medical images is not as pronounced as it is for transfer learning between natural images.

The main idea of our pre-training model comes from [1]. In their paper, they mainly used 3D images for training and trained different models for different modalities, in this thesis, we mainly trained 2D images and fused different modalities in the pre-training set, which really realized "model genesis".

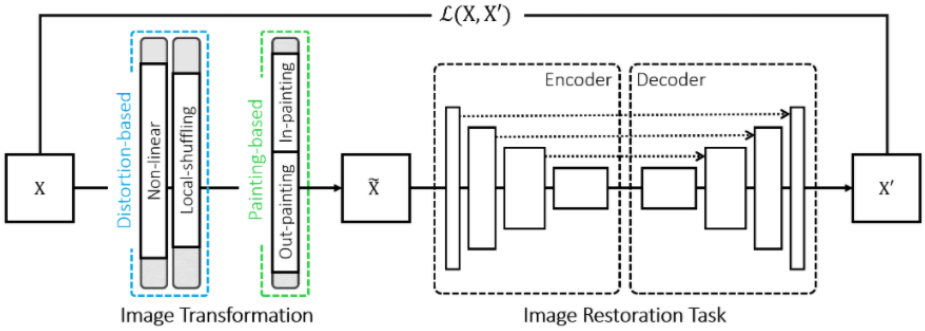

Figure 4.1: Model Genesis: unified self-supervised learning framework[1]

4.1.1.1 pre-text tasks for self-supervised learning

1. **Non-Linear:** The pixel values in CT are very much related to the structure of the human body. Consider the Fig.4.2, which reflects the pixel values in CT, called Hounsfield Units, which correspond to the air, fat, water, bones, and vital organs in the human body[1].

Substance		Hounsfield units (HU)
Air		-1000
Fat		-120 to -90
Water		0
Bone	Cancellous	+300 to +400
	Cortical	+1800 to +1900
Parenchyma	Lung	-700 to -600
	Kidney	+20 to +45
	Liver	+54 to +66
	Lymph nodes	+10 to +20
	Muscle	+35 to +55

Figure 4.2: Hounsfield Units in CT [1]

For individual human structures, the equivalent is that we have labels that are accurate to the pixel level, and the ability to reduce the pixels to the correct range based on the appearance of this tissue, which gives the model a strong ability to learn.

2. **Local-Shuffling:** Identify a small frame, disrupt the position of the pixels within the frame, and repeat the process several times to get a transformed image[1], the images were recovered through a self-supervised learning approach.

3. **In-/Out-Painting:** The out-painting means masking the edges of the original image, and let the model fill in, the in-painting means to cover some area in the middle of the original image and let the model fill in[1].

4. **Combine:** The above introduces four different image transformations, which can be easily combined to give an input that undergoes multiple transformations to get an output that is more robust to the model[1].

4.2 Transfer learning

4.2.1 Transfer learning for LUNA16

This section mainly classify the LUNA16 data and train three different models, 1) training from scratch, 2) training by using our pre-trained weights, 3) training by using imagenet weights, and compare the AUC of the three models.

In the next section the training results of the three different models will be describe in detail.

4.2.1.1 Data

Luna16 dataset contains 10 subfolders, 888 patients, each subfolder includes about 88 patients. The slice thickness of the scans ranges between 0.8 and 2.5 mm. The voxel values of the CT-scans are encoded on the Hounsfield Unit (HU) scale.

4.2.1.2 Patch extraction

I have tried 2 methods to extracted patches, the first method is directly extracted 2-D patches according to the candidates_v2.csv,the csv file I got is from LUNA16 challenge website, the csv file that contains an extended set of candidate locations for the 'false positive reduction' track. Since the coordinates of the candidates are given in world coordinates, I did a transform from world coordinates to voxel coordinates. Also I define some normalized planes to extract views from the candidates. The patches size I extracted is 64 * 64. The training dataset has extremely high negative to positive ratio (753418 : 1157).

The 2nd method I tried is to extracted 3-D cubes(64*64*64) based on the candidates_v2.csv, after extracted cubes, I random extracted 2-D patches from planes of a cube.

both of this 2 methods get similar result. Here are some examples of patches: Fig.4.3 are the negative patches, and Fig.4.4 are the positive patches.

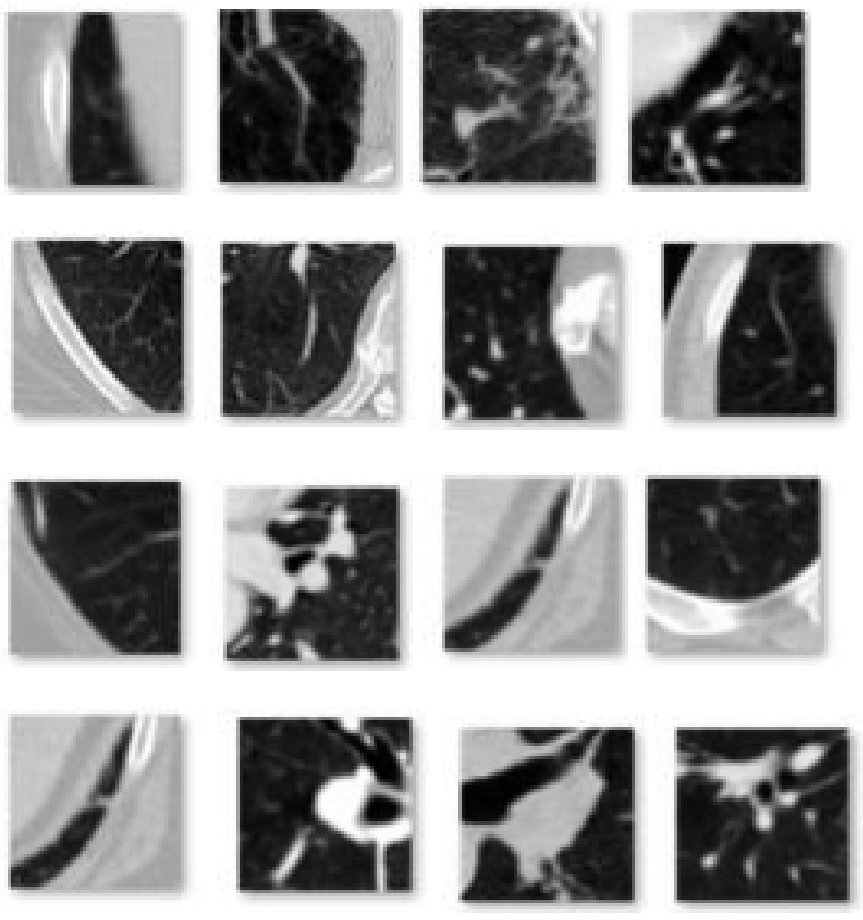

Figure 4.3: LUNA16 negative patches

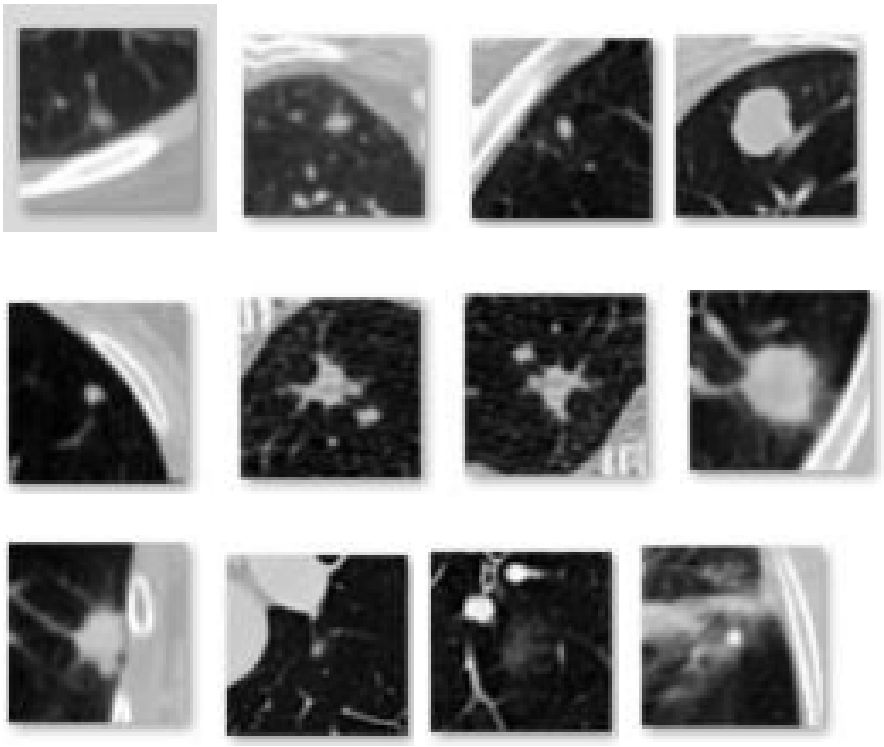

Figure 4.4: LUNA16 positive patches

4.2.1.3 Data augmentation

The number of categories in the dataset is very unbalanced: negative patches to positive patches ratio is (753418 :1157). Since the data is very unbalanced and it is difficult to train a good classifier, I first do a data augmentation on the data for the positive patches, and down-sample the negative patches, keeping 10% negative images per folder.

I applied common data augmentations for the positive patches: For each candidate, I performed random zooming [0.9,+1.1], random rotation [200 ,+200], and translation [-1 mm,+1 mm], vertical and horizontal flip. Finally increasing 8 times the positive patches. In the end, the negative to positive ratio is 82:18. Here are some examples of augmentation:

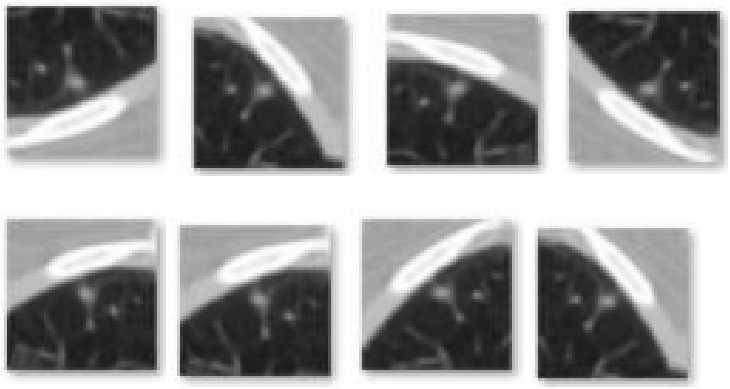

Figure 4.5: LUNA16 patches agumentation

4.2.1.4 Data pre-processing

I use the same approach as what I did on pre-training phase, by setting the MIN_BOUND to -1000 and MAX_BOUND to 1400. The current value range is [-1024,2000]. And values arbitrarily greater than 1400 do not need to be considered, by using the formula I pre-processed the image to the 0-1 interval.

$$image = \frac{image - MIN_{BOUND}}{MAX_{BOUND} - MIN_{BOUND}}$$

Figure 4.6

4.2.1.5 CNN configuration

The main structure of our pre-trained model is:

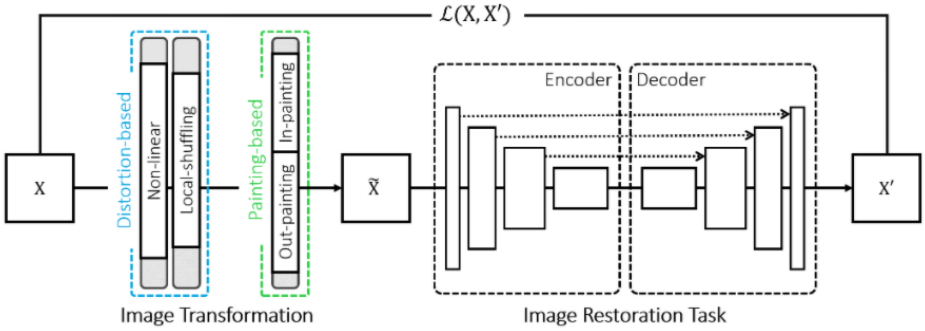

Figure 4.7: pre-trained model[1]

4.2.1.6 CNN configuration

After pre-training, a 2D U-Net-like structure is obtained, including encoder and decoder. Since it is solving a classification problem, this study decided to get the part of the encoder and use it as a feature extractor for the downstream classification task, after this encoder we added the GlobalAveragePooling2D, Flatten ,Dense layer with L2 regularizers and the output Dense layers, and to prevent overfitting, we add the Dropout layer after the Flatten layer and dense layer and set it to 0.5.

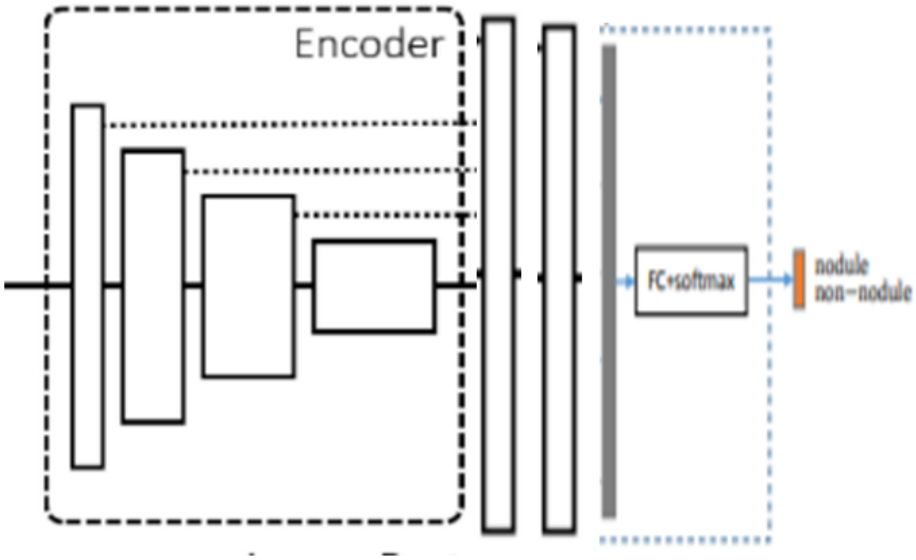

Figure 4.8: CNN configuration of classification model

4.2.1.7 Training and Testing

In the training and testing of the network, the method used is 5-fold cross validation. The dataset is divided into 10 subsets of approximately equal size. Within these subsets, 8 subsets are used to train the network. The remaining 2 subset is then used to test the results. This process was repeated 5 times.

4.2.2 results by using Unet VGG16 encoder Batch Normalization

In order to find the best hyperparameters, I set the learning rates 0.1, 0.01, 1e-3, 1e-4, 5e-4, and 1e-5 as the initial learning rates, tested the dropout rate at 0.5 and 0.2 respectively, and compared the results with and without the 'warm up' mechanism(by using 'warm up' mechanism, the training is divided into two phases, the whole point of first training the head separately was to avoid training layers randomly initialized with layers fine-tuned, using the initial learning rate only train the output layer, in the 2nd phase, I fine-tuned all the layers).

The best parameters this experiment used are below:

	Optimizers	Loss	Epoch	Batch size	Input shape
Training from scratch	SGD(lr=0.001) ReduceLROnPlateau Factor 0.01	binary_crossentropy	18	32	64*64
Pre-trained weights	SGD(lr=0.001) ReduceLROnPlateau Factor 0.01	binary_crossentropy	13	32	64*64
ImageNet weights	SGD(lr=0.001) ReduceLROnPlateau Factor 0.01	binary_crossentropy	13	32	64*64

Figure 4.9: different parameters used in the training

I calculate the AUC after each epoch, and I trained 3 models for each model I trained 5 times.

4.2.2.1 Training from scratch

I plotted the model ROC and Loss after the 5-folder cross-validation.As shown in the figure below.

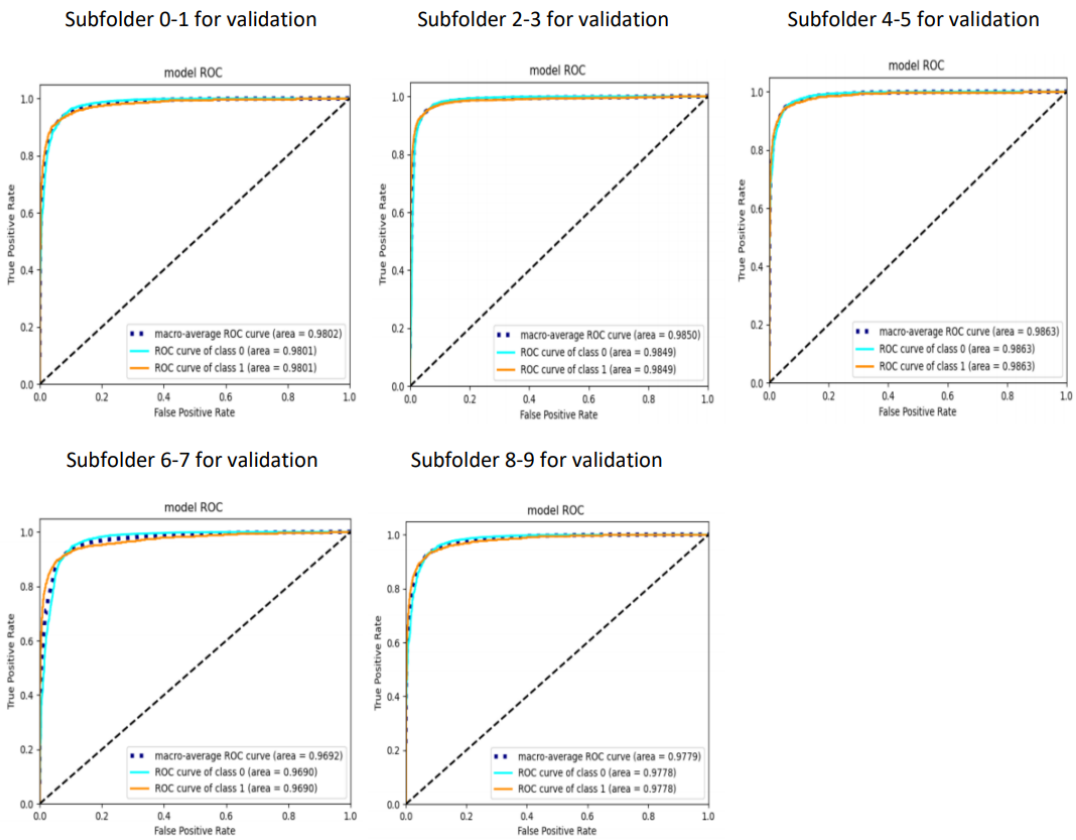

Figure 4.10: model ROC after 5-folder cross-validation

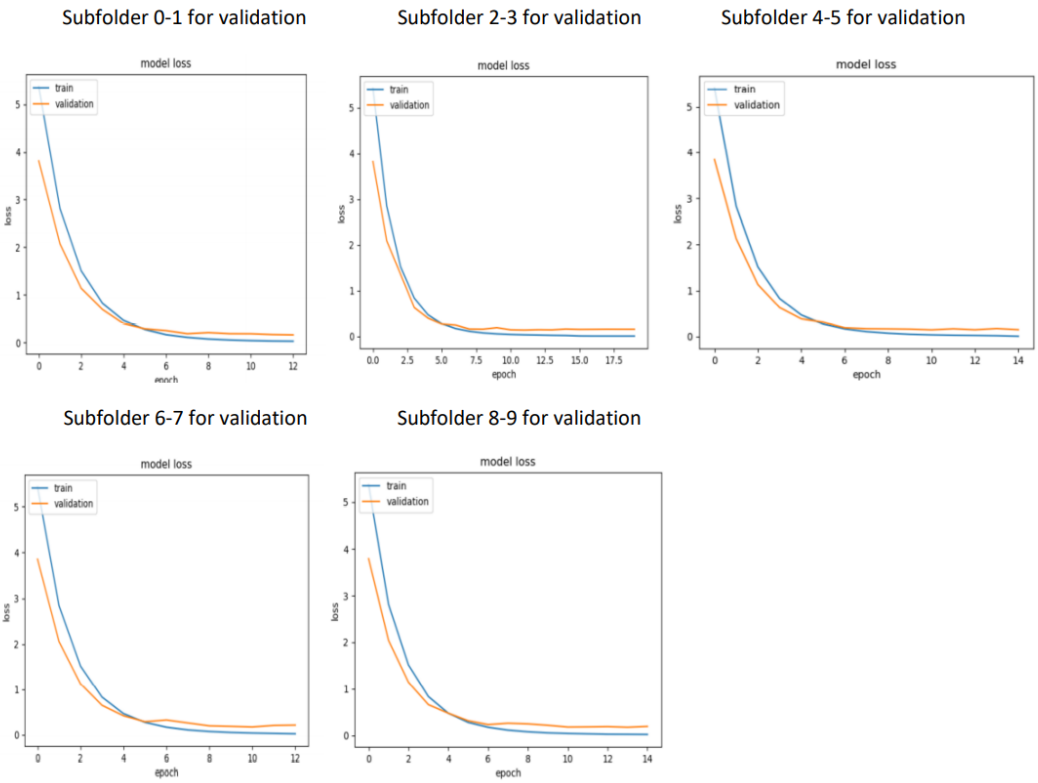

Figure 4.11: model loss after 5-folder cross-validation

For better visualization, we summarize the results after 5-folder corss-validation as follows.

Subset	AUC
Subset0-1	0.9802
Subset2-3	0.9850
Subset4-5	0.9863
Subset6-7	0.9692
Subset8-9	0.9779
AVG	0.98172

Figure 4.12: A summary of AUC after 5-folder cross validation training from scratch

4.2.2.2 Transfers by using our pre-trained weights

I plotted the model ROC and Loss after the 5-folder cross-validation.As shown in the figure below.

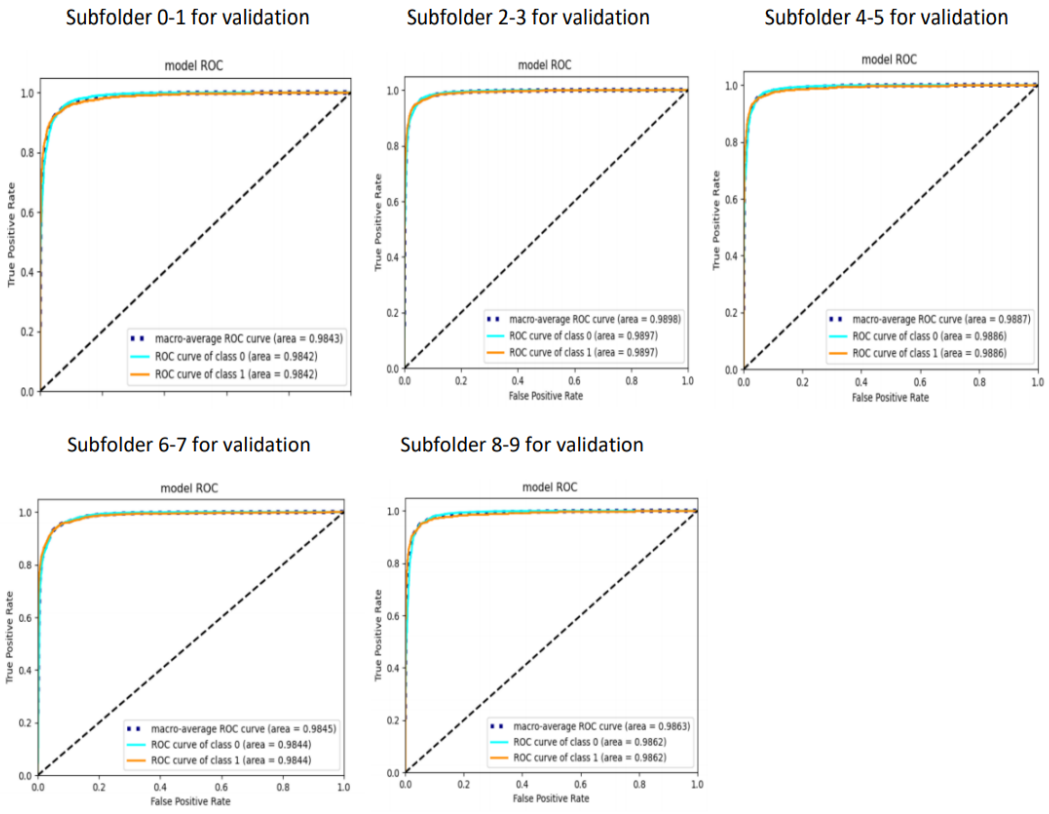

Figure 4.13: model ROC after 5-folder cross-validation

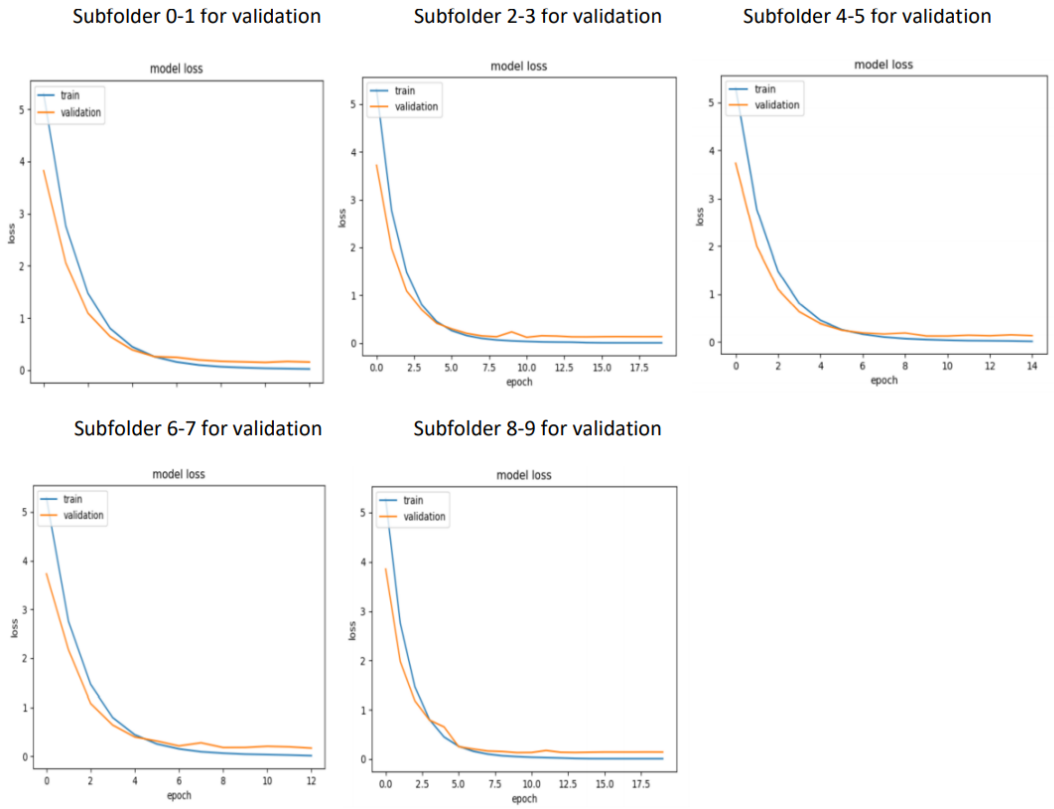

Figure 4.14: model loss after 5-folder cross-validation

We summarize the results after 5-folder corss-validation as follows.

Subset	AUC
Subset0-1	0.9843
Subset2-3	0.9898
Subset4-5	0.9887
Subset6-7	0.9845
Subset8-9	0.9863
AVG	0.98672

Figure 4.15: A summary of AUC after 5-folder cross validation training by using our pre-trained weights

4.2.2.3 Transfers by using imagenet weights

I plotted the model ROC and Loss after the 5-folder cross-validation.As shown in the figure below.

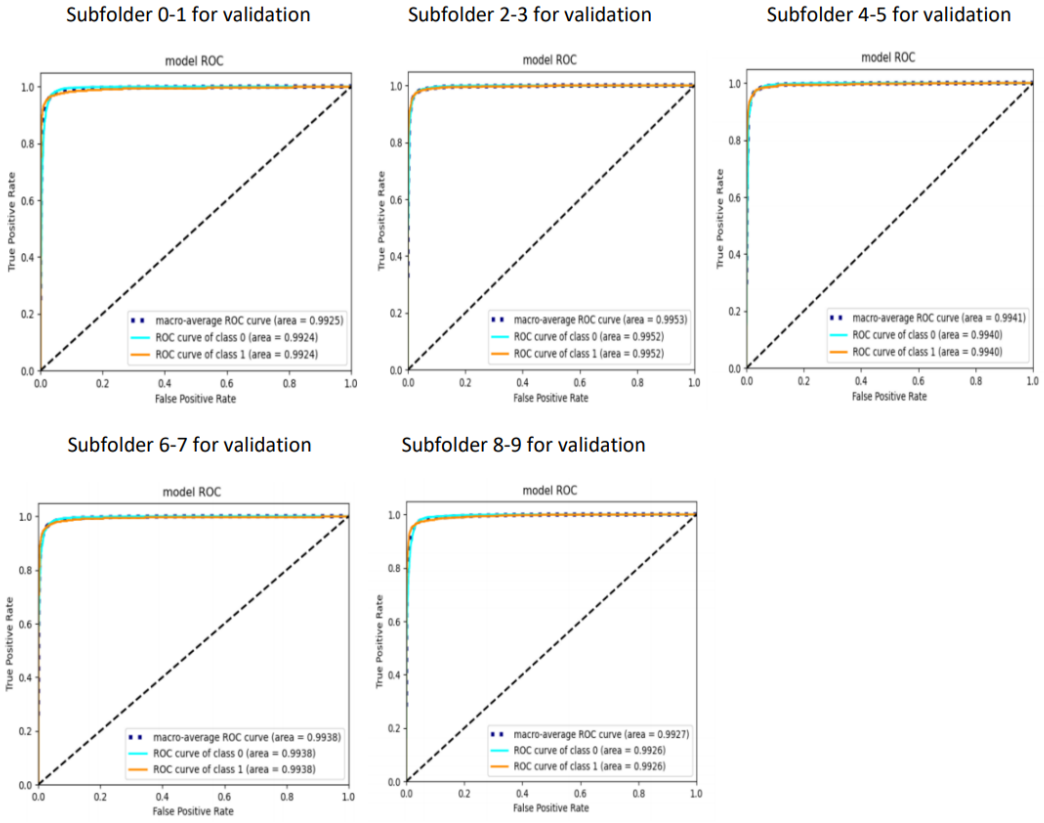

Figure 4.16: model ROC after 5-folder cross-validation

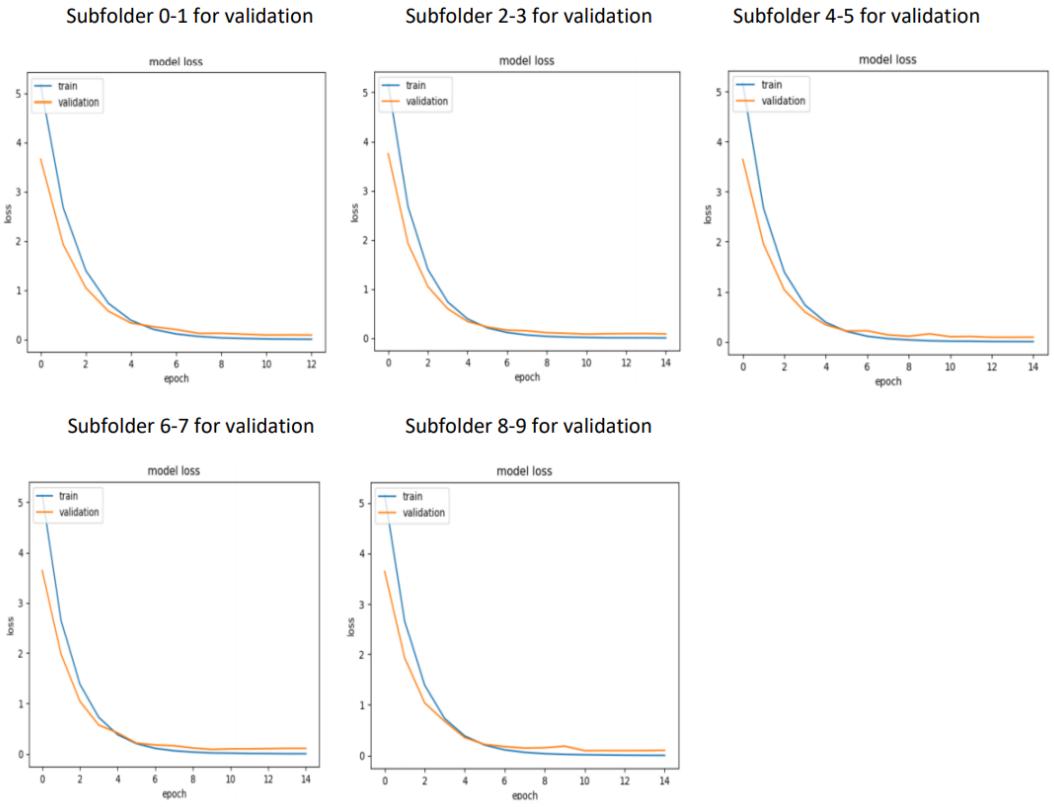

Figure 4.17: model loss after 5-folder cross-validation

We summarize the results after 5-folder corss-validation as follows.

Subset	AUC
Subset0-1	0.9925
Subset2-3	0.9953
Subset4-5	0.9941
Subset6-7	0.9938
Subset8-9	0.9926
AVG	0.99366

Figure 4.18: A summary of AUC after 5-folder cross validation training by using Imagenet weights

4.2.2.4 Summary of the three models

Model	AUC
Scratch	0.98172
Pre-trained weights	0.98672
ImageNet weights	0.99366

Figure 4.19: AUC summary of three models I

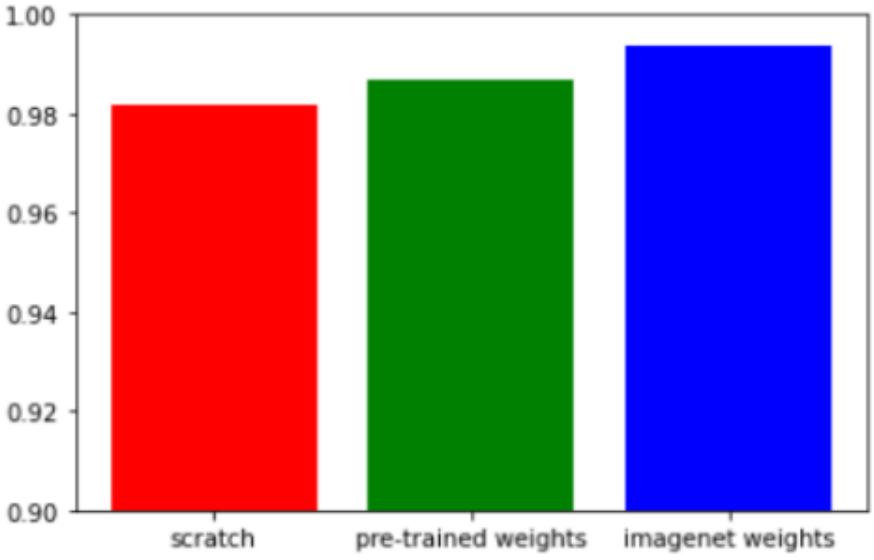

Figure 4.20: AUC summary of three models II

4.2.3 results by using Unet Resnet50 encoder Batch Normalization

The way of select the best parameters is same with what I described before. The best parameters the experiment used are below:

Optimizers: SGD with a initial learning rate 0.001, momentum 0.9, we set ReduceLROnPlateau factor=0.1, patience=3.

Loss: binary_crossentropy

Epoch: 10

Batch size: 32

input size: 64*64

Calculate the AUC after each epoch.

This thesis trained 3 models and each model trained 5 times.

4.2.3.1 Training from scratch

I plotted the model ROC and Loss after the 5-folder cross-validation.As shown in the figure below.

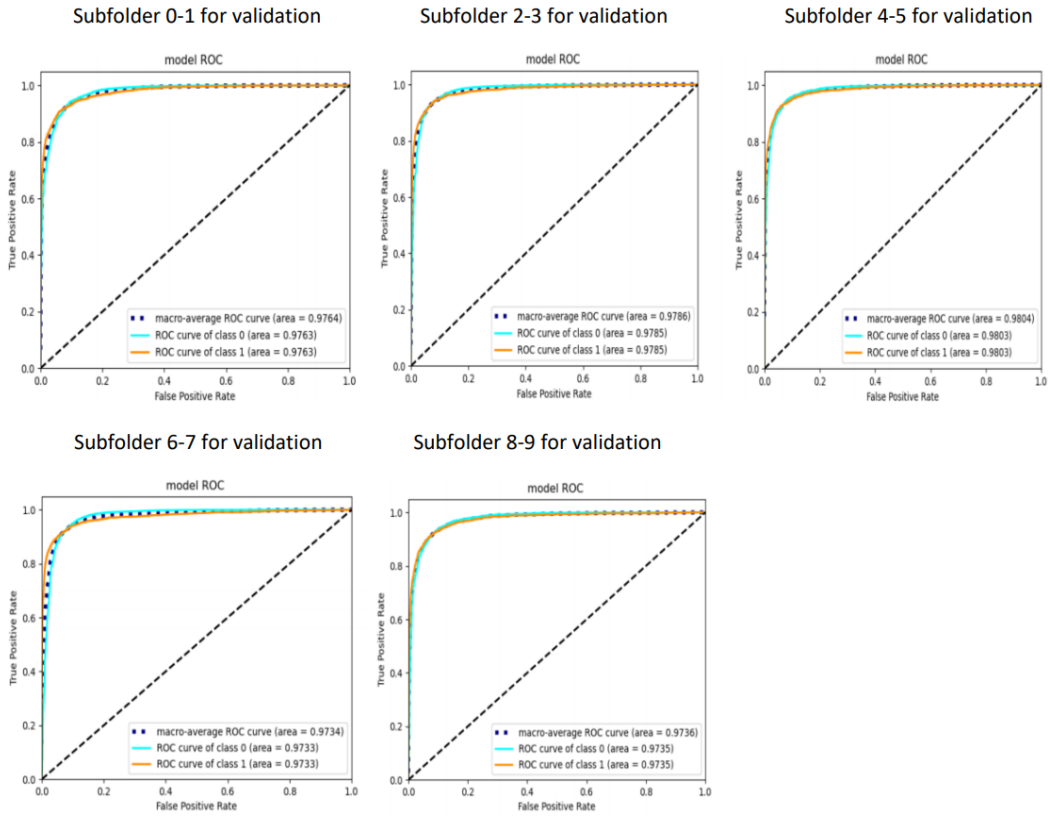

Figure 4.21: model ROC after 5-folder cross-validation

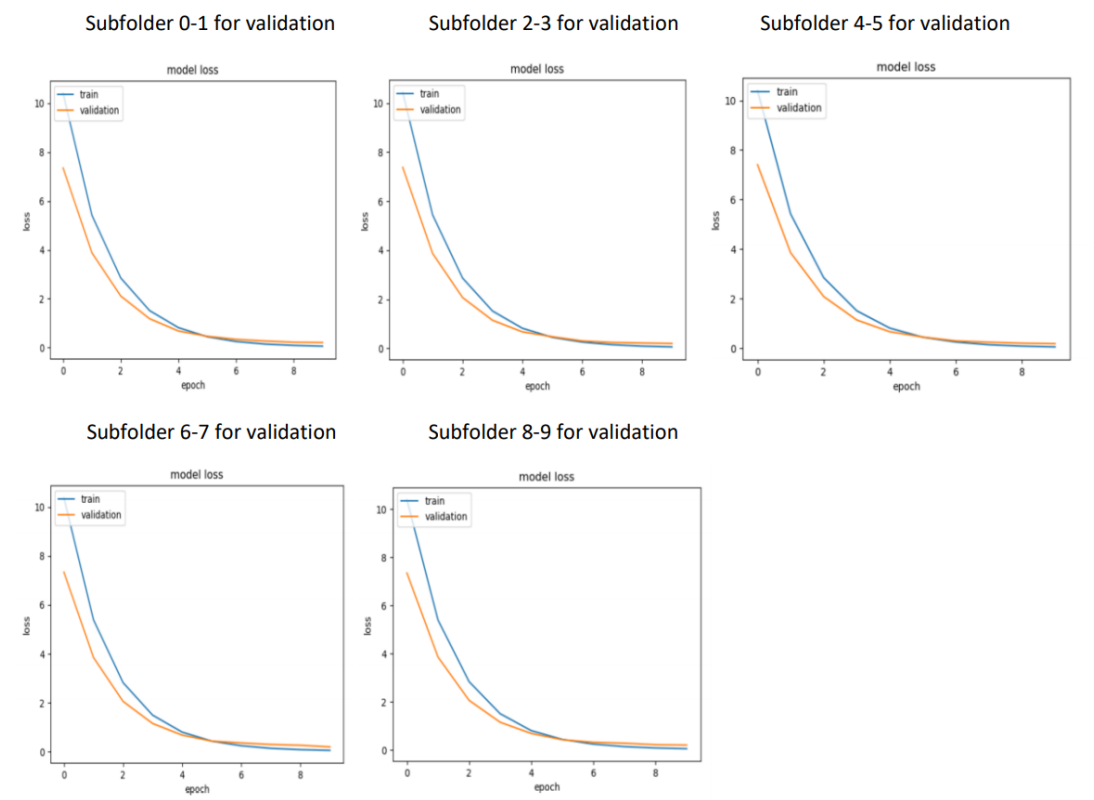

Figure 4.22: model loss after 5-folder cross-validation

We summarize the results after 5-folder corss-validation as follows.

Subset	ACU
Sub0-1	0.9764
Sub2-3	0.9786
Sub4-5	0.9804
Sub6-7	0.9734
Sub8-9	0.9736
AVG	0.97648

Figure 4.23: A summary of AUC after 5-folder cross validation training from scratch

4.2.3.2 Transfers by using our pre-trained weights

I plotted the model ROC and Loss after the 5-folder cross-validation.As shown in the figure below.

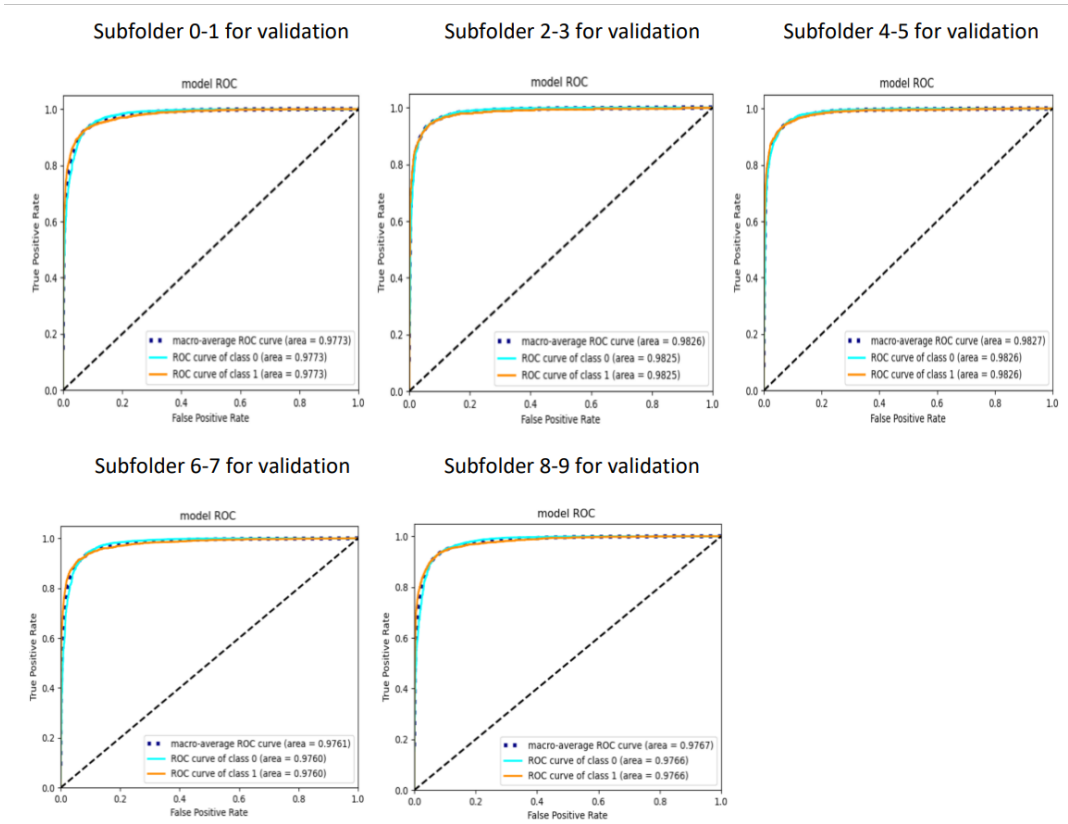

Figure 4.24: model ROC after 5-folder cross-validation

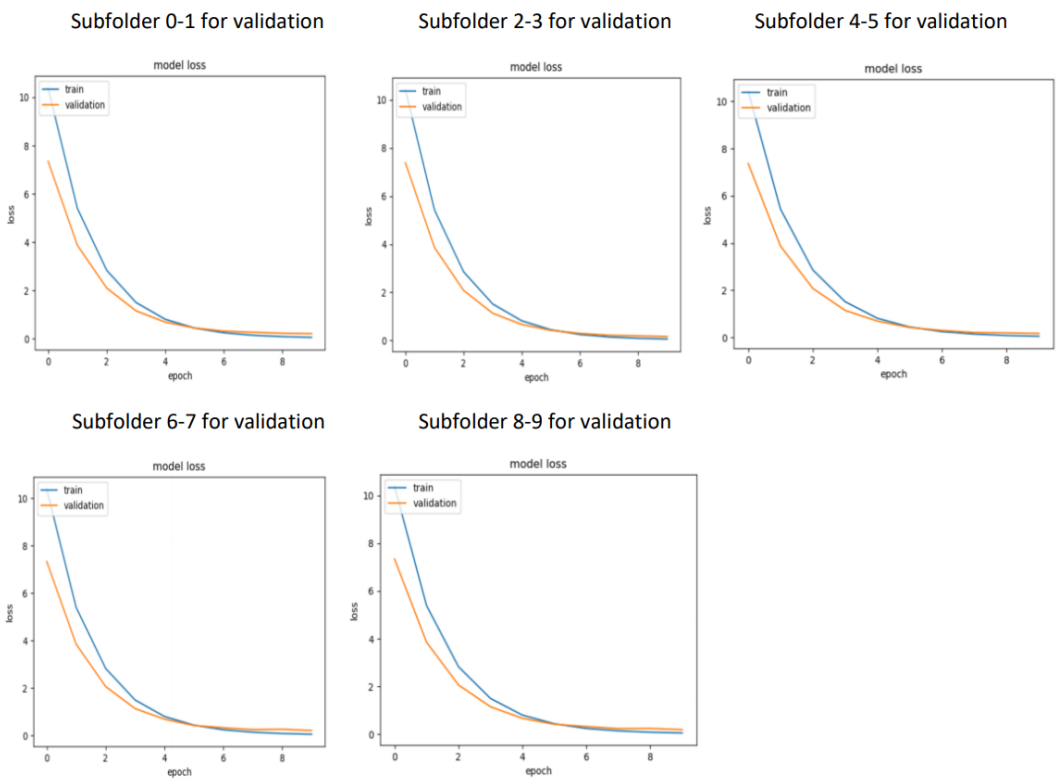

Figure 4.25: model loss after 5-folder cross-validation

We summarize the results after 5-folder corss-validation as follows.

Subset	ACU
Sub0-1	0.9773
Sub2-3	0.9826
Sub4-5	0.9827
Sub6-7	0.9761
Sub8-9	0.9767
AVG	0.97908

Figure 4.26: A summary of AUC after 5-folder cross validation training by using our pre-trained weights

4.2.3.3 Transfers by using imagenet weights

I plotted the model ROC and Loss after the 5-folder cross-validation.As shown in the figure below.

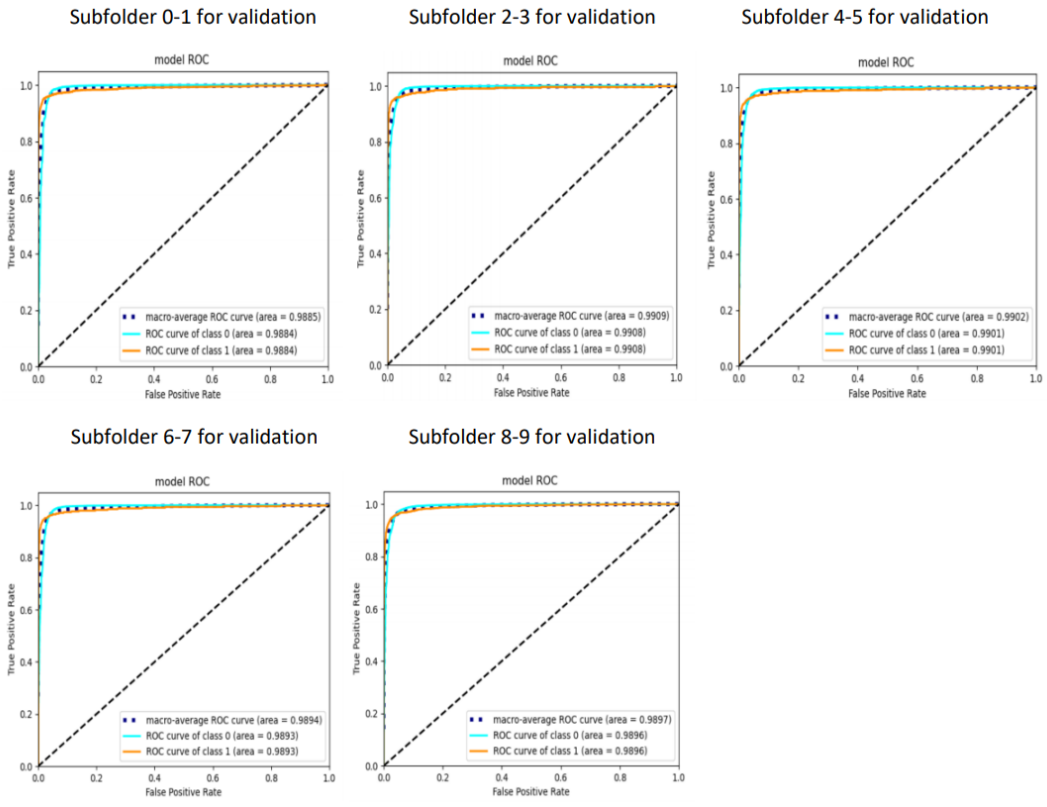

Figure 4.27: model ROC after 5-folder cross-validation

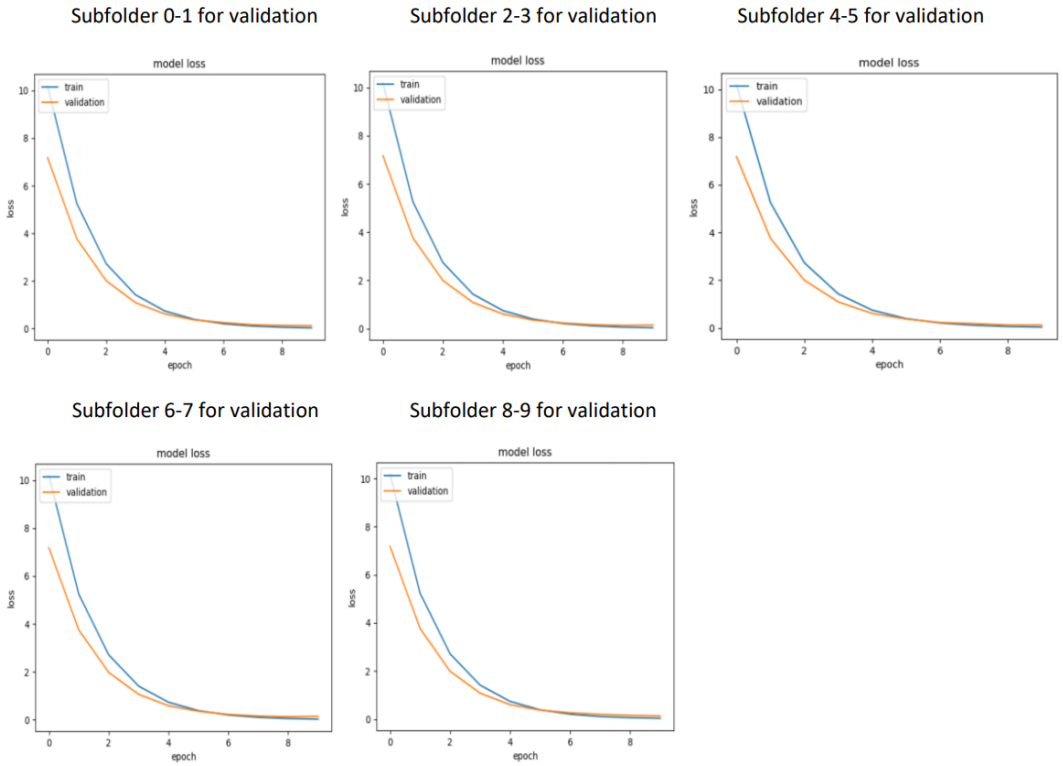

Figure 4.28: model loss after 5-folder cross-validation

We summarize the results after 5-folder corss-validation as follows.

Subset	ACU
Sub0-1	0.9885
Sub2-3	0.9909
Sub4-5	0.9902
Sub6-7	0.9894
Sub8-9	0.9897
AVG	0.98974

Figure 4.29: A summary of AUC after 5-folder cross validation training by using Imagenet weights

4.2.3.4 Summary of the three models

Model	AUC
Scratch	0.97648
Pre-trained weights	0.97908
Imagenet weights	0.98974

Figure 4.30: AUC summary of three models I

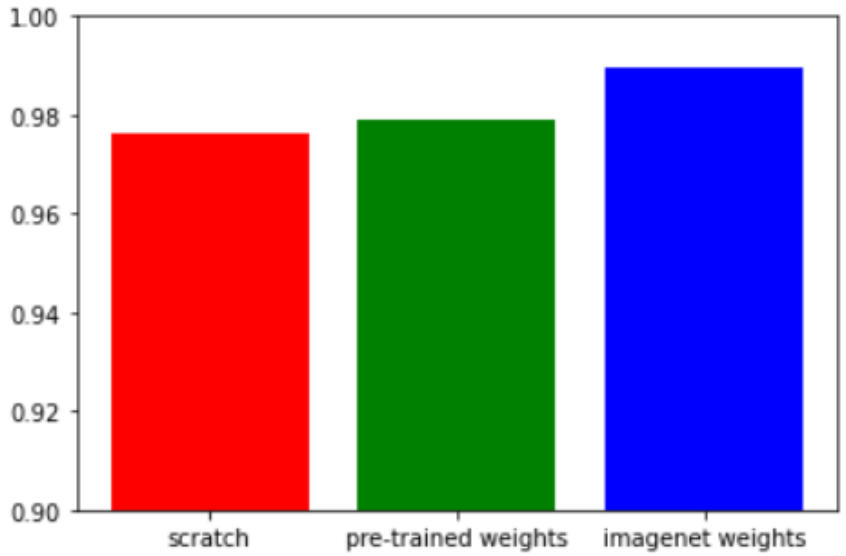

Figure 4.31: AUC summary of three models II

4.2.4 Transfer learning for MURA

4.2.4.1 Data

The MURA dataset contains X-ray images of seven different body parts, and in this experiment mainly used the shoulder dataset. Shoulder dataset contains positive(4168),Negative(4211) images, and automatically separates the training and validation sets.

4.2.4.2 Method and Data augmentation

This experiment was to do image classification on the image level, and the size of the image was 224*224. Before the image was input to the network, data augmentation was first performed. The data augmentation this experiment performed is in the following way: rotation_range: 20, zoom_range: [0.9,+1.1], horizontal_flip: True ,vertical_flip: True. After data augmentation the data distribution become Negative 29477 Positive 29176. Some examples after data augmentation:

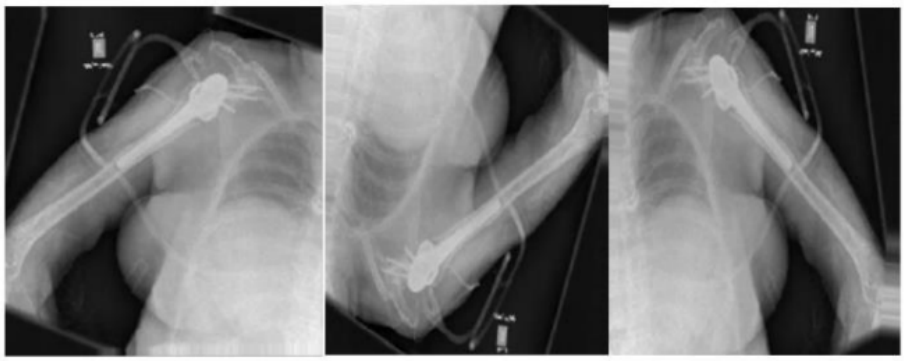

Figure 4.32: examples after data augmentation

4.2.4.3 Data normalization

Before feed the images into the network, I did data normalization, this can be achieved by dividing all pixel values by the 255, we have normalized the image pixel distribution in the [0-1] range, also in this experiment the input size of the image has changed to (224,224).

4.2.4.4 Training and validation

This experiment used the default validation set as the final validation set, using the data after data augmentation as the training set, and trained three models (training from scratch, transfer by using pre-trained weights, transfer by using imagenet weights), in order to avoid the randomness of the experimental results, for the best model in the experiment, I trained it 2 times(because of the time-consuming), and calculate the average of AUC, the evaluation metrics of the experiment is AUC and accuracy, finally I plot the ROC curve as the final results.

4.2.5 results by using Unet Resnet50 encoder Batch Normalization

In order to find the best hyperparameters, I set the learning rates 0.1, 0.01, 1e-3, 1e-4, 5e-4, and 1e-5 as the initial learning rates, tested the dropout rate at 0.5 and 0.2 respectively, and compared the results with and without the 'warm up'mechanism(by using 'warm up' mechanism,the training is divided into two phases, the whole point of first training the head separately was to avoid training layers randomly initialized with layers fine-tuned, using the initial learning rate only train the output layer, in the 2nd phase, fine-tune all the layers).

The best parameters this experiment used are below:

	Optimizers	Loss	Epoch	Batch size	Input shape
Training from scratch	SGD(lr=0.001) Factor 0.01	binary_crossentropy	12	16	224*224
Pre-trained weights	SGD(lr=0.001) Factor 0.1	binary_crossentropy	Warm up: 2 epoch Fine-tune: 9 epoch	16	224*224
ImageNet weights	SGD(lr=0.001)	binary_crossentropy	Warm up: 2 epoch Fine-tune: 3 epoch	16	224*224

Figure 4.33: different parameters used in the training

4.2.5.1 Training from scratch

The best model's AUC is 0.7573

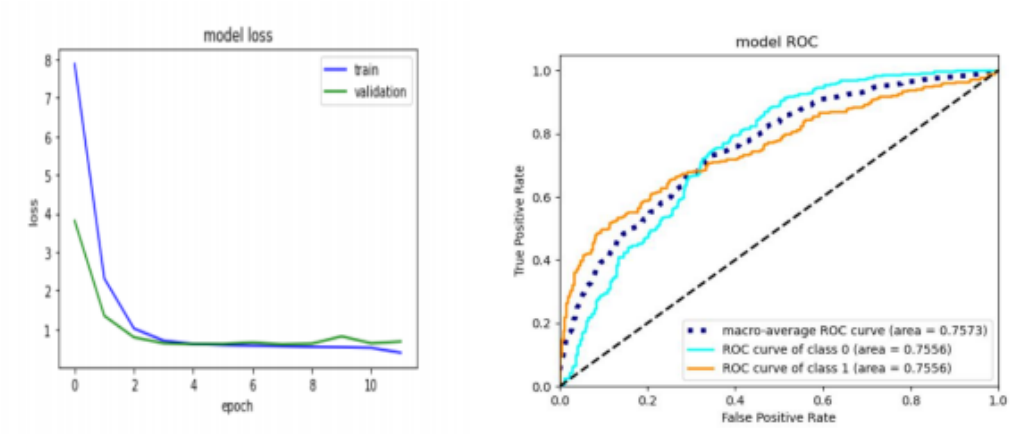

Figure 4.34: training from scratch model ROC and loss

4.2.5.2 Transfer by using pre-trained weights

The best model's AUC is 0.7791

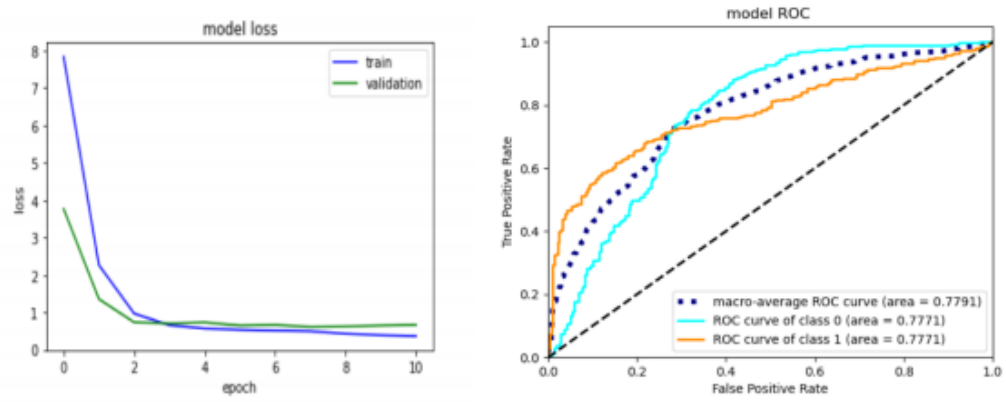

Figure 4.35: transfer by using pre-trained weights model ROC and loss

4.2.5.3 Transfer by using ImageNet weights

The best model's AUC is 0.8319

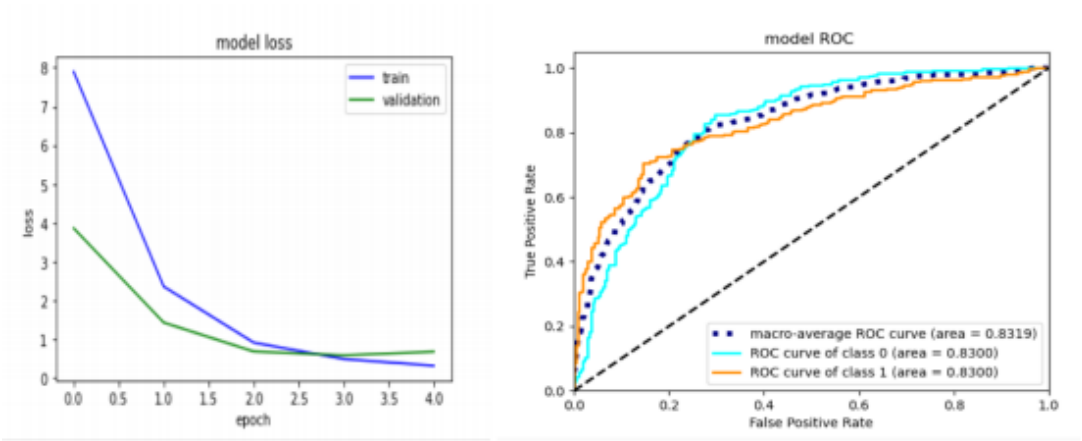

Figure 4.36: transfer by using ImageNet weights model ROC and loss

4.2.5.4 AUC comparison between 3 models

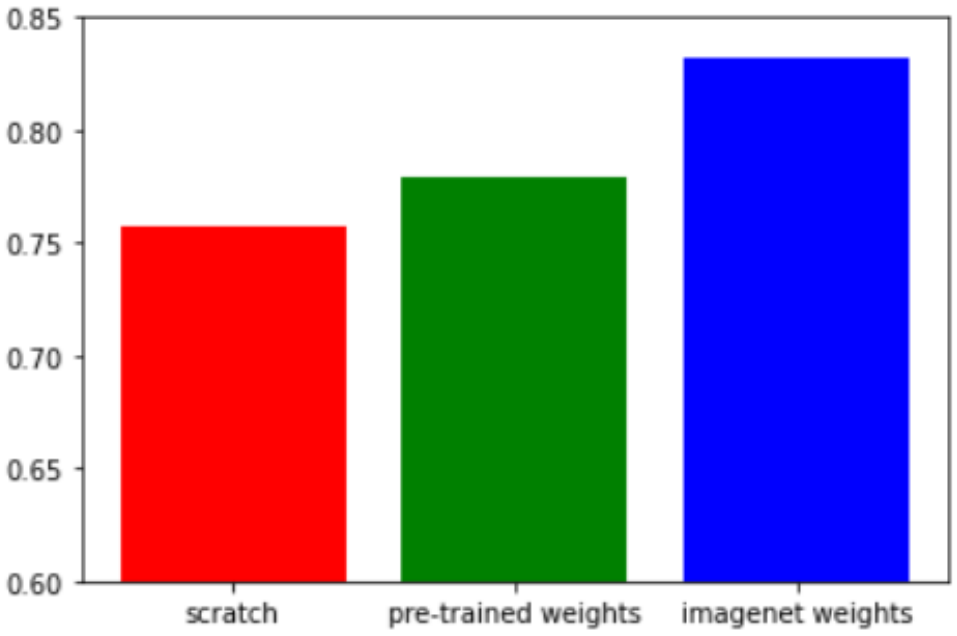

Figure 4.37: AUC comparison between 3 models

4.2.6 results by using Unet VGG16 encoder Batch Normalization

How to choose the best hyperparameters is described in the previous section. The best parameters this experiment used are below:

	Optimizers	Loss	Epoch	Batch size	Input shape
Training from scratch	SGD(lr=0.001) ReduceLROnPlateau Factor 0.01	binary_crossentropy	15	16	224*224
Pre-trained weights	SGD(lr=0.001) ReduceLROnPlateau Factor 0.1	binary_crossentropy	12	16	224*224
ImageNet weights	SGD(lr=0.001) ReduceLROnPlateau Factor 0.01	binary_crossentropy	5	16	224*224

Figure 4.38: different parameters used in the training

4.2.6.1 Training from scratch

Model loss show bellow, and the final AUC is 0.7897.

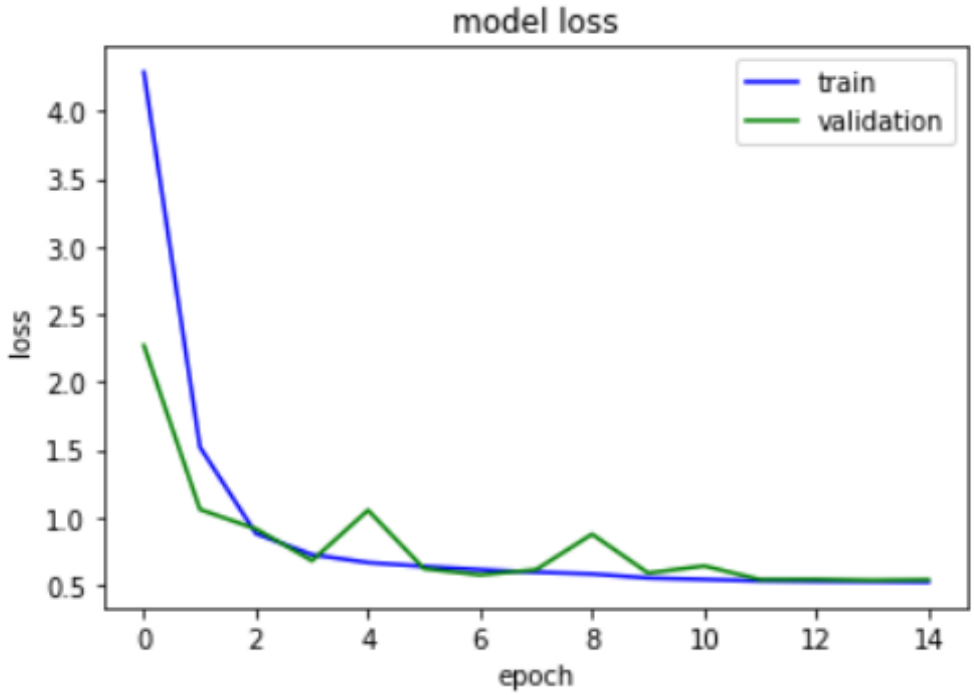

Figure 4.39: training from scratch model loss

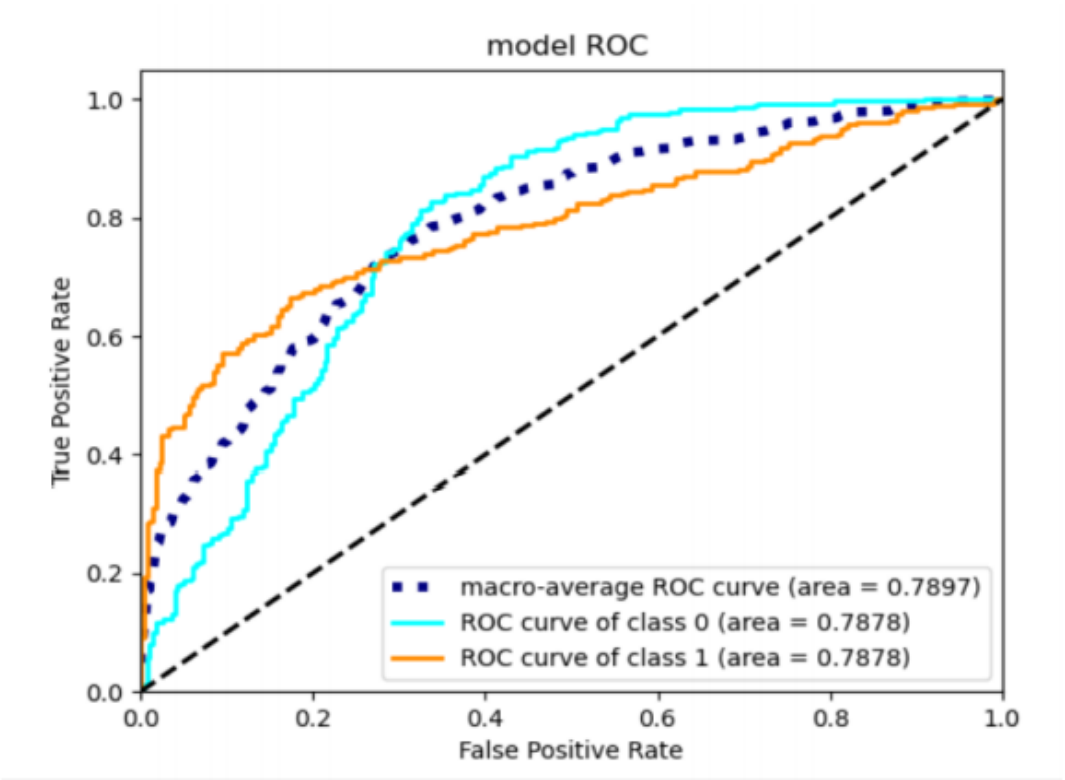

Figure 4.40: training from scratch model ROC

4.2.6.2 Transfer by using pre-trained weights

The final results were obtained by performing two experiments on the best model and calculating the mean of the AUC. The final AUC(average 0.8291).

Experiment 1:

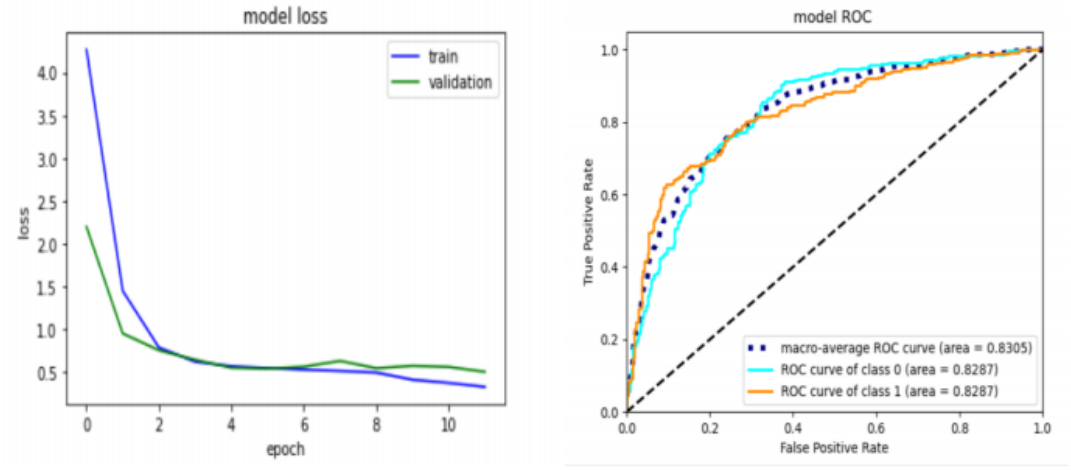

Figure 4.41: transfer by using pre-trained weights model ROC and loss

Experiment 2:
Model AUC: 0.8277

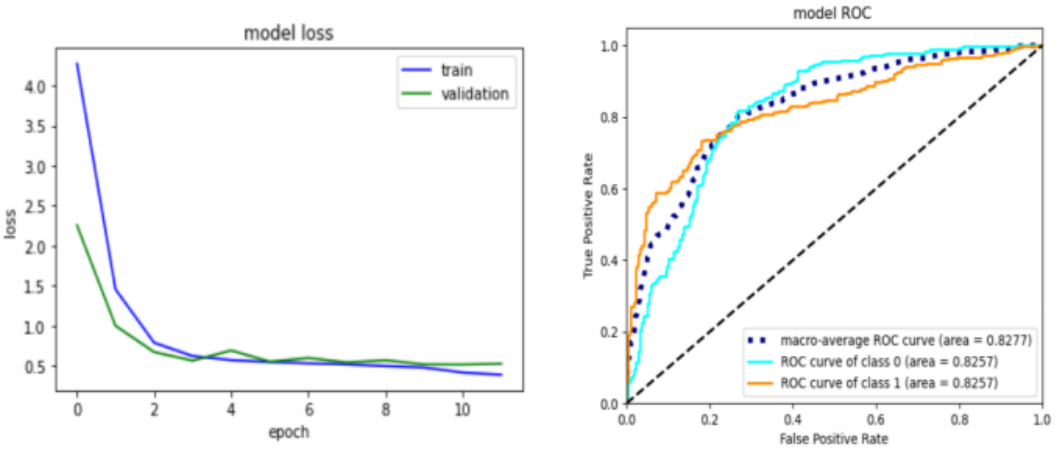

Figure 4.42: transfer by using pre-trained weights model ROC and loss

4.2.6.3 Transfer by using ImageNet weights

The final results were obtained by performing two experiments on the best model and calculating the mean of the AUC. The final AUC(average 0.8287).

Experiment 1:

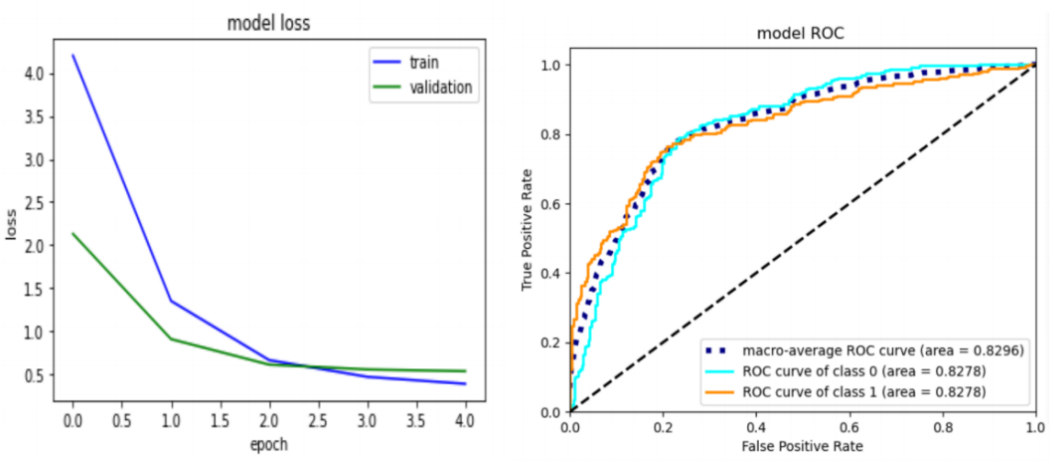

Figure 4.43: transfer by using ImageNet weights model ROC and loss

Experiment 2 :

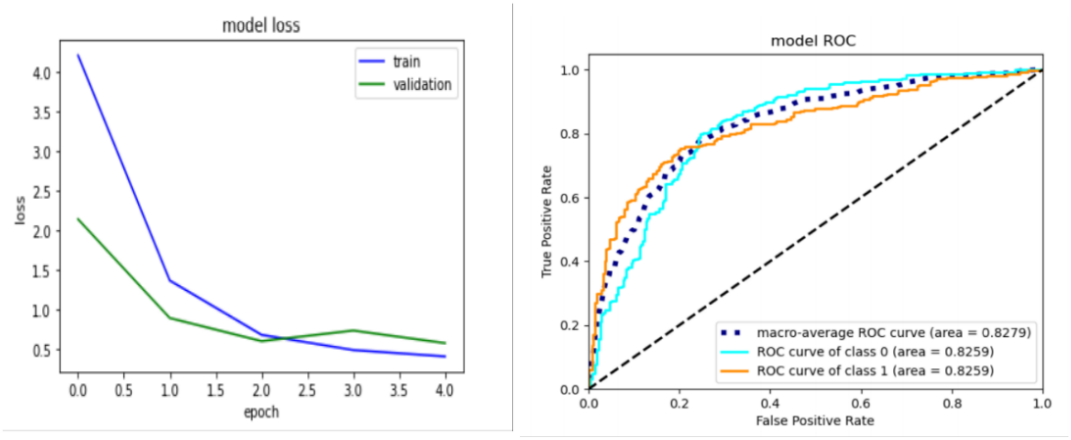

Figure 4.44: transfer by using ImageNet weights model ROC and loss

4.2.6.4 AUC comparison between 3 models

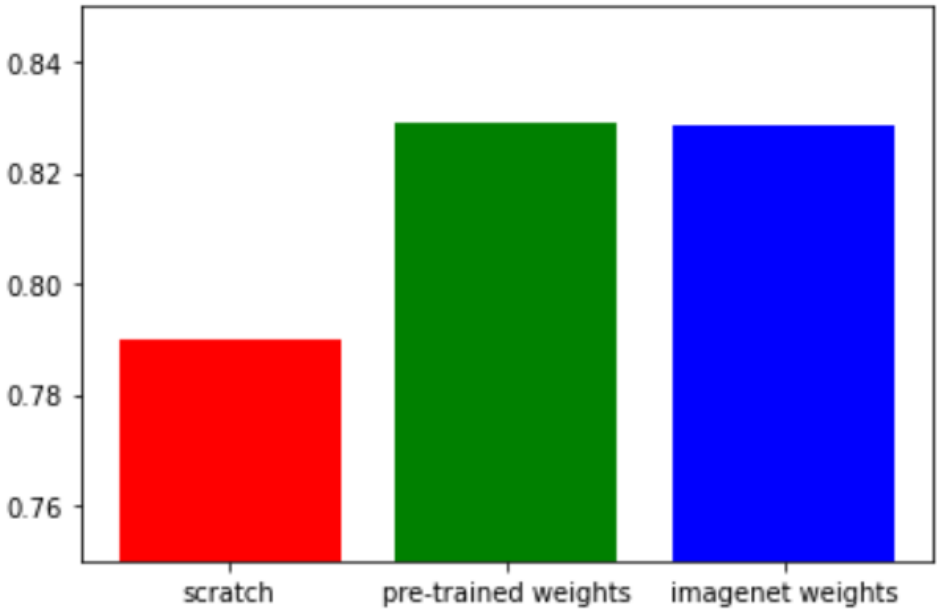

Figure 4.45: AUC comparison between 3 models

Chapter 5

Conclusions and future works

5.1 Conclusions

This thesis exploit a novel deep learning method self-supervised learning by training on 21 datasets of medical images including different modalities, body parts, and diseases.It then transfer the pre-training weights to downstream tasks for classification tasks.

The problem of lack of labeling in medical image datasets was well solved by the self-supervised learning approach, and the transfers of the pre-trained weights to downstream classification tasks also worked better than the model trained from scratch. In our experiments, we transfered this pre-trained weights on LUNA16 dataset and MURA dataset to do image classification task. The model is trained with Resnet50 and VGG16 backbone respectively.

From the four sets of experiments conducted on the above two datasets, we summarize the results for each of the best parameters. The final results shows on Fig.5.1. and Fig.5.2.

Dataset	Backbone	Metric	Scratch	Pre-trained	ImageNet
LUNA16	Resnet50	AUC	0.97648	0.97908	0.98975
LUNA16	Vgg16	AUC	0.98172	0.98672	0.99366
MURA	Resnet50	AUC	0.7573	0.7791	0.8319
MURA	Vgg16	AUC	0.7897	0.8291	0.8287

Figure 5.1: Results of all experiments

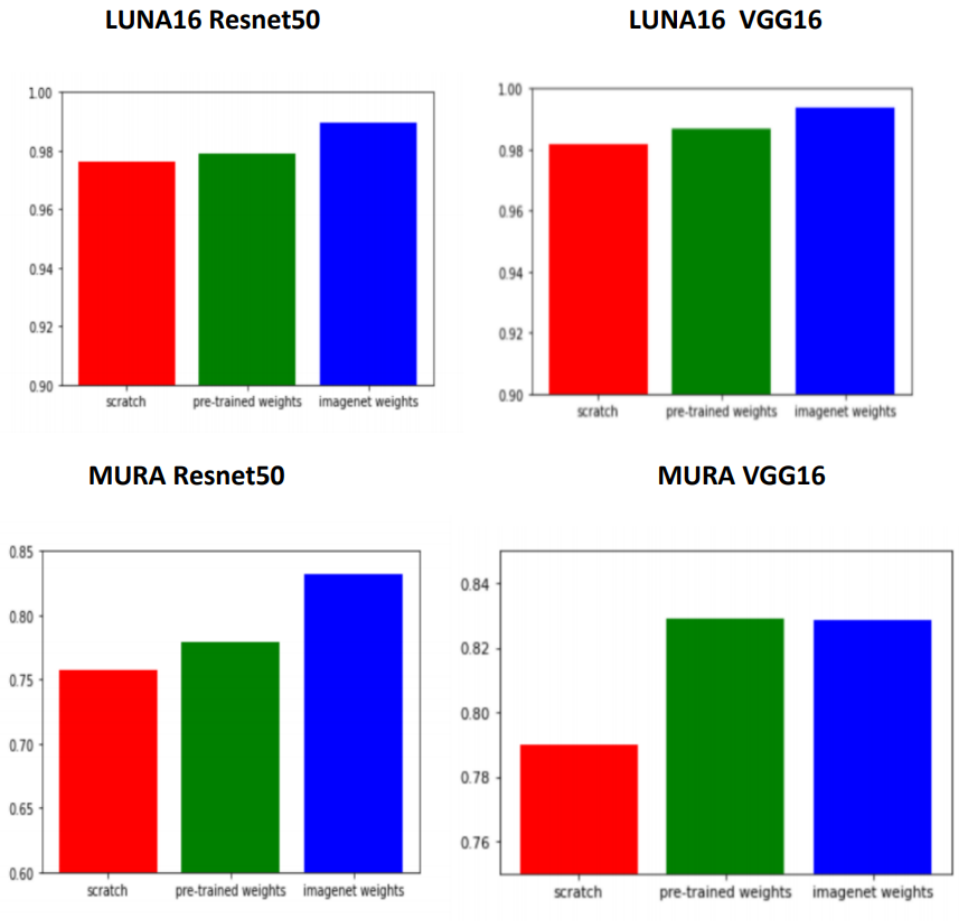

Figure 5.2: Results of all experiments

We conclude that when using the VGG16 backbone for the image level classification task on MURA dataset, the AUC of transfer learning using our pre-trained model can get a similar result as transfer learning by using ImageNet.

Also in the other three conditions, using our pre-trained weights on transfer learning for classification tasks were more accurate than training the network from scratch.Although the LUNA16 dataset is not present in the pre-trained dataset, using transfer learning with the weights of the pre-trained model is still helpful in the medical image classification task.

However, using our pre-trained network to train a model does not converge faster than ImageNet transfers. And in all other cases, the AUC of the model is lower than ImageNet transfers.

5.2 Future works

From the experiments, we can see that when using the resnet50 backbone for training, the results are not satisfactory. Although the results are better than the network trained from scratch, there is still a gap with the state of the art Imagenet transfers.

Our previous hypothesis was that the tranfer ability of our pre-trained model would exceed that of ImageNet transfers, because ImageNet contains a very large variety of images from nature, whereas our pre-trained model contains only three different modalities of medical images, thus making it more task specific, but we did not achieve the expected results.

During my experiments, I found that when I do transfer learning with ImageNet, the loss tends to converge very fast, and I can get a good result with very little epoch training, while using our pre-trained weights, our loss converges faster than the training from scratch, but still converges slower than with ImageNet transfers.

I think it's most likely that we didn't get a good weights when we did the pre-training by using self-supervised learning approach, or even that the training wasn't completed thoroughly.

In the future, we will review the pre-training network again, and retrain the transfers with a new pre-trained network, use more datasets(CAD-PE,ChexPert) for different downstream tasks(segmentation,detection) and achieve real "model genesis".

Bibliography

[1] Zongwei Zhou, Vatsal Sodha, Md Mahfuzur Rahman Siddiquee, Ruibin Feng, Nima Tajbakhsh, Michael B Gotway, and Jianming Liang. «Models genesis: Generic autodidactic models for 3d medical image analysis». In: *International Conference on Medical Image Computing and Computer-Assisted Intervention*. Springer. 2019, pp. 384–393 (cit. on pp. 1, 14, 41–43, 47).

[2] Fabian Isensee et al. «nnu-net: Self-adapting framework for u-net-based medical image segmentation». In: *arXiv preprint arXiv:1809.10486* (2018) (cit. on pp. 2, 40).

[3] Keiron O'Shea and Ryan Nash. «An introduction to convolutional neural networks». In: *arXiv preprint arXiv:1511.08458* (2015) (cit. on pp. 5, 7).

[4] Yann LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. «Gradient-based learning applied to document recognition». In: *Proceedings of the IEEE* 86.11 (1998), pp. 2278–2324 (cit. on p. 7).

[5] Longlong Jing and Yingli Tian. «Self-supervised visual feature learning with deep neural networks: A survey». In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* (2020) (cit. on pp. 9, 10).

[6] Chuen-Kai Shie, Chung-Hisang Chuang, Chun-Nan Chou, Meng-Hsi Wu, and Edward Y Chang. «Transfer representation learning for medical image analysis». In: *2015 37th annual international conference of the IEEE Engineering in Medicine and Biology Society (EMBC)*. IEEE. 2015, pp. 711–714 (cit. on p. 13).

[7] Jason Yosinski, Jeff Clune, Yoshua Bengio, and Hod Lipson. «How transferable are features in deep neural networks?» In: *Advances in neural information processing systems*. 2014, pp. 3320–3328 (cit. on p. 13).

[8] Min Chen, Xiaobo Shi, Yin Zhang, Di Wu, and Mohsen Guizani. «Deep features learning for medical image analysis with convolutional autoencoder neural network». In: *IEEE Transactions on Big Data* (2017) (cit. on p. 13).

[9] Takayasu Moriya, Holger R Roth, Shota Nakamura, Hirohisa Oda, Kai Nagara, Masahiro Oda, and Kensaku Mori. «Unsupervised segmentation of 3D medical images based on clustering and deep representation learning». In: *Medical Imaging 2018: Biomedical Applications in Molecular, Structural, and Functional Imaging*. Vol. 10578. International Society for Optics and Photonics. 2018, p. 1057820 (cit. on p. 13).

[10] Deepak Pathak, Philipp Krahenbuhl, Jeff Donahue, Trevor Darrell, and Alexei A Efros. «Context encoders: Feature learning by inpainting». In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2016, pp. 2536–2544 (cit. on p. 14).

[11] Gustav Larsson, Michael Maire, and Gregory Shakhnarovich. «Colorization as a proxy task for visual understanding». In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2017, pp. 6874–6883 (cit. on p. 14).

[12] Mehdi Noroozi and Paolo Favaro. «Unsupervised learning of visual representations by solving jigsaw puzzles». In: *European Conference on Computer Vision*. Springer. 2016, pp. 69–84 (cit. on p. 14).

[13] Mehdi Noroozi, Hamed Pirsiavash, and Paolo Favaro. «Representation learning by learning to count». In: *Proceedings of the IEEE International Conference on Computer Vision*. 2017, pp. 5898–5906 (cit. on p. 14).

[14] Wenjia Bai, Chen Chen, Giacomo Tarroni, Jinming Duan, Florian Guitton, Steffen E Petersen, Yike Guo, Paul M Matthews, and Daniel Rueckert. «Self-supervised learning for cardiac mr image segmentation by anatomical position prediction». In: *International Conference on Medical Image Computing and Computer-Assisted Intervention*. Springer. 2019, pp. 541–549 (cit. on p. 14).

[15] Roth Holger, Le Lu, Ari Seff, Kevin M Cherry, Joanne Hoffman, Shijun Wang, and Ronald M Summers. *A new 2.5 D representation for lymph node detection in CT. The Cancer Imaging Archive. 2015* (cit. on p. 17).

[16] Amber L Simpson et al. «A large annotated medical image dataset for the development and evaluation of segmentation algorithms». In: *arXiv preprint arXiv:1902.09063* (2019) (cit. on p. 17).

[17] Sasank Chilamkurthy, Rohit Ghosh, Swetha Tanamala, Mustafa Biviji, Norbert G Campeau, Vasantha Kumar Venugopal, Vidur Mahajan, Pooja Rao, and Prashant Warier. «Development and validation of deep learning algorithms for detection of critical findings in head CT scans». In: *arXiv preprint arXiv:1803.05854* (2018) (cit. on p. 17).

[18] A Emre Kavur et al. «CHAOS Challenge–Combined (CT-MR) Healthy Abdominal Organ Segmentation». In: *arXiv preprint arXiv:2001.06535* (2020) (cit. on p. 18).

[19] Ke Yan, Xiaosong Wang, Le Lu, and Ronald M Summers. «DeepLesion: automated mining of large-scale lesion annotations and universal lesion detection with deep learning». In: *Journal of Medical Imaging* 5.3 (2018), p. 036501 (cit. on p. 18).

[20] Nicholas Bien et al. «Deep-learning-assisted diagnosis for knee magnetic resonance imaging: development and retrospective validation of MRNet». In: *PLoS medicine* 15.11 (2018), e1002699 (cit. on p. 18).

[21] Pranav Rajpurkar et al. «Mura: Large dataset for abnormality detection in musculoskeletal radiographs». In: *arXiv preprint arXiv:1712.06957* (2017) (cit. on pp. 18, 19).

[22] Alexander Andreopoulos and John K Tsotsos. «Efficient and generalizable statistical models of shape and appearance for analysis of cardiac MRI». In: *Medical Image Analysis* 12.3 (2008), pp. 335–357 (cit. on p. 18).

[23] Xiaosong Wang, Yifan Peng, Le Lu, Zhiyong Lu, Mohammadhadi Bagheri, and Ronald M Summers. «Chestx-ray8: Hospital-scale chest x-ray database and benchmarks on weakly-supervised classification and localization of common thorax diseases». In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2017, pp. 2097–2106 (cit. on p. 18).

[24] Jun Cheng, Wei Huang, Shuangliang Cao, Ru Yang, Wei Yang, Zhaoqiang Yun, Zhijian Wang, and Qianjin Feng. «Enhanced performance of brain tumor classification via tumor region augmentation and partition». In: *PloS one* 10.10 (2015), e0140381 (cit. on p. 18).

[25] Daniel S Marcus, Tracy H Wang, Jamie Parker, John G Csernansky, John C Morris, and Randy L Buckner. «Open Access Series of Imaging Studies (OASIS): cross-sectional MRI data in young, middle aged, nondemented, and demented older adults». In: *Journal of cognitive neuroscience* 19.9 (2007), pp. 1498–1507 (cit. on p. 18).

[26] L Palumbo, P Bosco, ME Fantacci, E Ferrari, P Oliva, G Spera, and A Retico. «Evaluation of the intra-and inter-method agreement of brain MRI segmentation software packages: A comparison between SPM12 and FreeSurfer v6. 0». In: *Physica Medica* 64 (2019), pp. 261–272 (cit. on p. 19).

[27] Yuxin Wu and Kaiming He. «Group normalization». In: *Proceedings of the European conference on computer vision (ECCV)*. 2018, pp. 3–19 (cit. on pp. 38, 39).