LYU1706

Using Deep Learning for Breast Cancer Diagnosis

SUPERVISED BY PROF. LYU RUNG TSONG MICHAEL

LI, QI 1155062147

LI, WEI 1155062146

Table of Contents

1 Abstract

The recent years witnessed a rapid development of Artificial Intelligence (AI). In May 2017, AlphaGo, the well-known AI program that plays the board game Go, challenged Chinese grandmaster Ke Jie, the best Go player in the world ranking. Go is one of the most difficult games in the field of AI due to the large search space. Thanks to the development of deep learning, which consists of deep neural networks, and the evolution of computing capability of computers, especially evolution of General-purpose computing on graphics processing units (GPUS), AI in complicated problems such as Go became more and more possible. There are many research teams trying to apply AI to medical diagnosis. Stanford published a paper on Nature about diagnosis of skin cancer in February 2017. Utilizing deep learning algorithm, their model achieved an accuracy of 91%, which is almost the comparable to a human doctor. They let the deep neural network to figure out the common visual features of the disease. Other top universities are also putting resources on applying deep learning algorithms to a variety of diseases' diagnosis, e.g. lung nodules and breast cancer.

Take breast cancer as an example. The recent years witnessed a considerable increase in the number of breast cancer cases. To improve the long-term survival rate for patients, the key factors are early detection and accurate diagnosis. However, the mismatch between increasing patients and the lack of experienced pathologists brings a lot of challenges for accurate diagnosis of breast cancer. The imbalance of the medical resources in allocation also increase the chance of misdiagnose. Therefore, we will try to build a reliable computer program to help pathologist do breast cancer diagnosis faster and easier. While there are many automatic medical diagnosis attempts, few of them are targeted at pathologists with little artificial intelligence background. Pathologist may not understand terms describing an AI or statistics an AI produces. There exists possibility that pathologist cannot interpret a computer generated report very well. With such limitation, cooperating with AI may instead delay decision-making.

Therefore, we will try to implement a complete automated breast cancer diagnosis system. In this system, we will train a deep learning program which can give advice to pathologists, even if s/he do not know anything about AI. It is designed to be able to perform mammogram analysis or pathology analysis, and detect possible tumor location. A deliverable diagnosis and tumor positioning report will be generated at the end which can help them make a more accurate decision on diagnosis. This problem involves image classification, object detection and image caption together.

In term one, our primary objective is to build an accurate breast cancer histopathological image classification model, which is the very first and most important procedure in our system. We train and test our model with BreaKHis, a breast cancer histopathological image dataset available to every researcher. Experimental evidence shows that our proposed deep learning model can effectively classify histopathological images even if the image resolution in our task is higher than in other image classification tasks. We achieved pretty high accuracy which was up to 90% average. Results shows that deep learning in breast cancer diagnosis is promising. Finally, we also study different input preprocessing techniques.

2 Introduction

2.1 Motivation

Reviewing patient's biological tissue samples by a pathologist is a conventional method for many diseases diagnosis, especially for cancer such as breast cancer. However, reviewing samples are laborious and time-intensive, which may delay decision-making. The reviewing of pathology slides is a very complex task. Sometimes agreement in diagnosis for some forms of breast cancer can be as low as 48% [1]. The difficulty in diseases diagnosis by pathologists is inevitable because the pathologists need to review all slides per patient while each of slide is 10+ gigapixels when digitized at 40X magnification.

On the other hand, current automatic medical diagnosis attempts are not targeted at pathologists with little artificial intelligence background. Pathologist may not understand terms describing an AI or statistics an AI produces. There exists possibility that pathologist cannot interpret a computer generated report very well. With such limitation, cooperating with AI may instead delay decision-making. Therefore, we will try to implement a complete automated breast cancer diagnosis system.

2.2 Background

Since AlphaGo showed the possibility that AI can beat human in real world tasks [2], more and more people in universities [3] [4] [5] or industries are interested in AI for medical usage. The number of papers about AI diagnosis is growing exponentially.

2.2.1 Development of AI Classifiers

The classification problem is an important component in the field of deep learning. It is targeted on judging a new sample belongs to which predefined sample category, according to a train set containing certain

number of known samples. The classification problem is also called supervised classification, since all samples in train set are labeled, and all categories are predefined [6]. Classifier is one of the pattern recognition applications.

The most widely applied AI classifier is spam email filter, which classify each email into “regular” or “junk”. Generally speaking, each instance in the classification problem will be transform into a computer analyzable vector, which is usually called “features”. A feature can be an enumeration or a number.

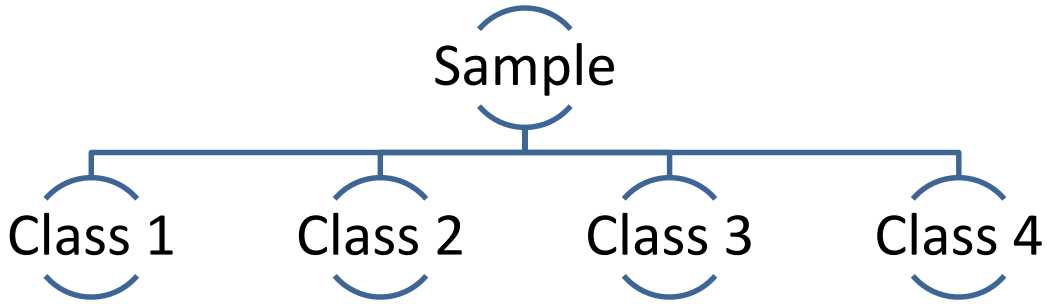

Figure 2.1 AI classification

Then the Naïve Bayes classifier was proposed in 1950s. It is a group of simple classifiers derived from the Bayes’ Theorem, assuming that all features in the samples are strongly independent. Since its publish, it has been widely researched. Things turned out that it performed well for text classification, with number of occurrence of words as features. It can do the aforementioned email classification task at a relatively low computation amount compared to more recent algorithms while still achieve acceptable accuracy [7]. With appropriate preprocessing, it is still competitive.

$$\text{posterior} = \frac{\text{prior} \times \text{liklihood}}{\text{evidence}}$$

$$p(C_k|F_1, \dots, F_n) = \frac{1}{Z} p(C_k) \sum_{i=1}^{n} (F_i|C_k)$$

$$\text{classify}(\text{f}_1, \dots, \text{f}_\text{n}) = \text{argmax}\ \ p(C = c) \prod_{i=1}^{n} p(\ F_i = \text{f}_\text{i}|C = c)$$

The Naïve Bayes classifier can have different assumptions for the underlying distribution of features. For continuous variables, we can assume they are under the classic Gaussian distribution. For text data, the standard assumption is multinomial distribution, where the number of occurrence of a word is taken into account. A simplified version is Bernoulli distribution, which only consider whether a word appears or not.

The Naïve Bayes classifier is much more extensible than other algorithms. Number of parameters it needs to learn is linear to number of features, therefore the training time complexity is also linear. Moreover, the training process has a well close-formed expression. For email classification problem, the number of parameters is merely the number of unique words in all emails. This avoid the expensive linear approximation many other classifiers use.

Later Support Vector Machine (SVM) was introduced by Vladimir Naumovich Vapnik and Alexey Yakovlevich Chervonenkis [8]. Given a train set, each sample is represented by a point the hyperspace. For SVM, samples are treated as p-dimensional vectors; SVM assumes that we can separate these points with a (p-1)-dimensional hyper plain. There may be may such hyper plain, and SVM will separate different categories with a hyper plain with as large margin as possible. Thus, we will get the hyper plain whose distance to nearest data points of two categories is maximized. This is also why it was named “Support Vector” machine.

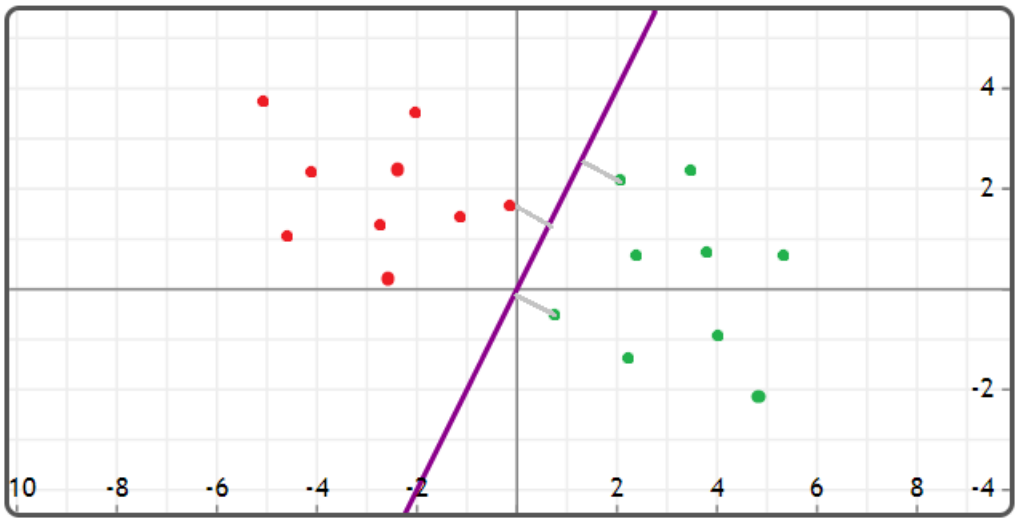

Figure 2.2 Support Vector Machine

SVM is usually a linear classifier. However, with some tricks called "Kernel trick", SVM can also do nonlinear classification. The main idea is, by mapping the original sample space to a higher dimensional space, the original non-linear separable set may become separable.

2.2.2 Development of Deep Learning

Deep learning is a subset of machine learning. It is a family of feature learning algorithms in the area of machine learning. Observation values can be represented in various ways, such as a vector containing RGB values of each pixel, or more abstractly a series of edges and areas [9]. It attempts to do highly abstract data computation with multiple process layers which may contain a complicated structure or non-linear mapping. In general, it is a boarder machine learning method, as it is not specific to any task. There are multiple deep learning frameworks already widely used, such as deep neural network, convolutional neural network and recursive neural network. Deep learning has been widely used in applications, including computer vision, natural language processing and bioinformatics, and achieves supreme results.

In 1989, Yann LeCun [10] proposed the deep learning mode. Through it could run, the computation cost was so large that the training took about three days. The very first deep learning attempt therefore failed going into real application. The trend of AI then shifted into Support Vector Machine. However, in 1992, Schmidhuber [11] proposed an effective algorithm to train neural networks. This algorithm treats each layer in the network as an unsupervised, and then tune its parameters with supervised back propagation algorithm. In experiment, it was shown that this training method can indeed improve the train speed of supervised learning.

The advantage of deep learning is that it uses effective unsupervised or Semi-supervised feature learning and layered feature extraction instead of man-powered feature extraction. The aim of feature learning is to seek for

better representation of data and to create better model to learn these representations from large-scale unlabeled dataset. The representation is like development of real neural network, and is based on the understanding of how information is processed and transmitted in neural-like systems [12].

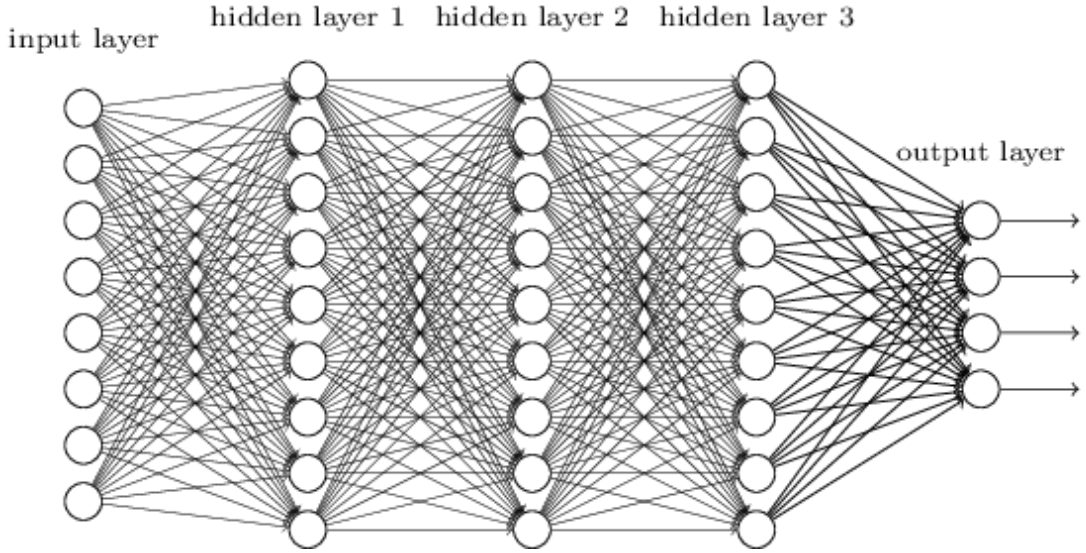

Figure 2.3 A deep neural network

The basis of deep learning is the distributed representation in machine learning. “Distributed” means the assumption that the observation is resulted from interaction between different factors. Furthermore, deep learning assumes that such interaction can be spliced into multiple layers, which means the multiple abstraction of the observed value. Different number of layers and different size of layers can be used to represent different degree of abstraction. This idea of layered abstraction indicates that higher-level concepts are learned from lower-level concepts. This structure is usually constructed with greedy algorithm, which helps the machine to learn more significant features. Many deep learning methods are unsupervised algorithms, which enables deep learning to be applied to unlabeled data. This is a great advantage over other algorithms. The amount of available unlabeled data is much larger than labeled ones; unlabeled data is also cheaper to acquire.

What even more encouraged researchers is General-Purpose computing on Graphics Processing Units (GPGPU). The development of more powerful hardware and increase in available data made deeper neural networks realizable. In 2009, Nvidia stepped into the area of deep learning and

started promoting its GPU. It was confirmed that the involvement of GPU can increase the training speed by more than 100 times. Since GPU is quite suitable for matrix/vector computation in deep learning algorithm, a GPU can reduce the time required from weeks to days.

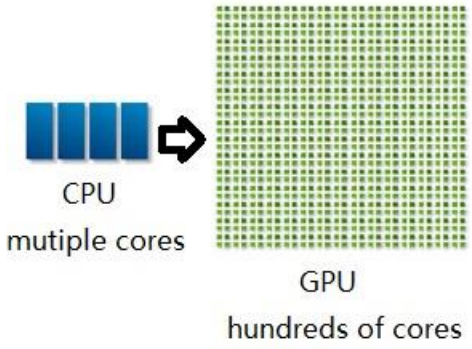

Figure 2.4 From CPU to GPU

Since the emerge of deep learning, it has become one part of the most advanced systems in various areas, especially in computer vision and speed recognition. On standard verification datasets such as Cifar 10, experiments showed that deep learning can improve recognition accuracy. A deep learning method, convolution neural network, processed about 10% to 20% checks in US [13]. Due to the development of deep learning, the year 2010 witnessed a bunch of the very first industrial speech recognition products [14].

2.2.3 Development of Deep Learning for Medical Images

In the area of medical image proceeding, deep learning is becoming more and more attractive. The recent development in deep learning has achieved a great leap. Generally speaking, research on deep learning for medical images is mainly focused on four aspects: structures detection, segmentation, labeling and captioning, and computer aided detection or diagnosis.

Structure detection is one of the most important steps in medical image process. Pathologists generally accomplish this task by recognizing some anatomical feature in the image. Though the success of deep learning in this area mainly depends on how many anatomical feature the algorithm

can extract. The recent trend indicates deep learning is mature enough to solve real world problems. Shin et al. [15] proved deep learning in computer vision applicable for medical images. On top of this, they detected multiple organs in a series of MRI images. Meanwhile, Roth et al. [16] presented a method to detect organ at certain body part. They trained their deep neural network with 4298 images and achieved an error rate of 5.9%.

Segmentation is the process of dividing a digital image into many sub-images [17]. A segment is a set of pixels, and therefore is also called hyper pixel. The aim of image segmentation is to simplify or alter the representation of the image so that it becomes more easy to understand or analyze. Segmentation is usually used to locate objects or edges in the image. More precisely, segmentation is a process to label each pixel in the image, which makes pixels with the same label have a similar visual feature, such as color, brightness or texture. Moeskops et al. [18] designed a multiple-scale CNN for accurate tissue segmentation, using multiple patch sizes and multiple convolution kernel sizes to gain multiple scale information of each pixel, and achieved accuracy from 82% to 91%. Zhang et al. [19] tested four CNN on the task of brain tissue segmentation. Their experiment uses three convolution layers and a fully connected layer, and proved CNN significantly better than traditional methods.

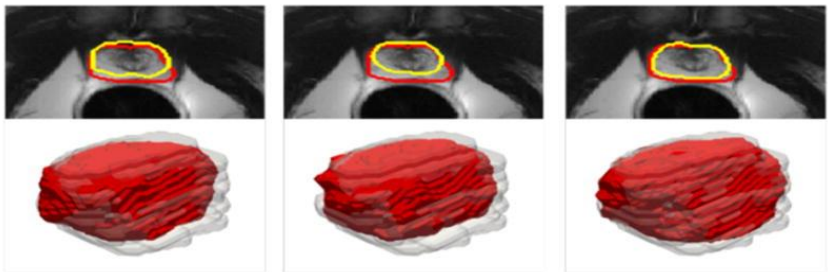

Figure 2.5 Typical Segmentation

Labeling and captioning is the most widely used way to describe contents in an image. It is the classic classification problem in the area of medical images. Continuous effort is being put in to ensure disease-specific auto labeling. Inspired by neural networks for regular images, some research [20] [21] introduced RNN together with latest advance in computer vision to caption chest radiographs in certain contexts. The authors used image

captions in public available dataset to train the CNN. To avoid large error, many normalization techniques were applied. Then the network was used to describe the situation of detected disease.

Computer aided detection or diagnosis involves finding or locating abnormalities and suspicious area, and then alert clinicians. The main aim of computer aided detection is to increase the detection rate of infected area and to decrease false negative due to observer's mistake. Though it is considered a mature area in medical images, deep learning further improved performance in many applications and enabled some design that was impossible in the past. Traditionally, computer detection requires a preprocessed candidate region and manpower to extract features such as shape or statistics in the region; inly after then the features can be feed into the classifier. However, the advantage of feature learning is the core of the new developments. Deep learning can learn the hierarchical features from the dataset independently instead of depending handcrafted features specially targeted for certain area of knowledge. It soon proved to be the most advanced technology. Ciompi et al. [22] trained CNN with predefined OverFeat as feature extractor, and showed that CNN is feasible to provide useful feature description in lung images. Gao et al. [23] trained the model from the very beginning. They solved the overfitting problem by randomly cropping or jittering the original image, and then feed the sub images into the model. Finally, the model was able to classify patches into normal, fibrosis and other four abnormal classes.

Due to the prosperity in research, more and more commercial attempts is being conducted recently. Startups entering the medical AI area is increasing. From 2012 to 2016, investments in medical AI increases from 20 cases per year to 70 cases per year. More than 100 large companies are trying to apply deep learning in order to decrease time to provide aids to patient and to automatically diagnosis disease with medical images. IBM Watson Group is supporting a research to screen cancer patients with an affordable procedure. They are trying to make deep learning suitable for production. Other startups include SkinVision, Flatiron Health and Entopsis [24].

2.3 Objective

Deep learning has a natural advantage in features learning, which means that it has a potential to be applied to this problem mentioned above. Therefore, we will try to implement a complete automated breast cancer diagnosis system. In this system, we will train a deep learning program which can give advice to pathologists, even if s/he do not know anything about AI.

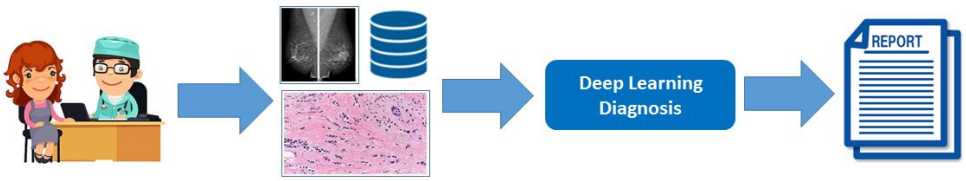

Figure 2.6 Our Diagnosis System

This project involves image classification, object detection and image caption together. It is designed to be able to perform mammogram analysis or pathology analysis, and detect possible tumor location. A deliverable diagnosis and tumor positioning report will be generated at the end which can help them make a more accurate decision on diagnosis. The whole system will have the following functionalities:

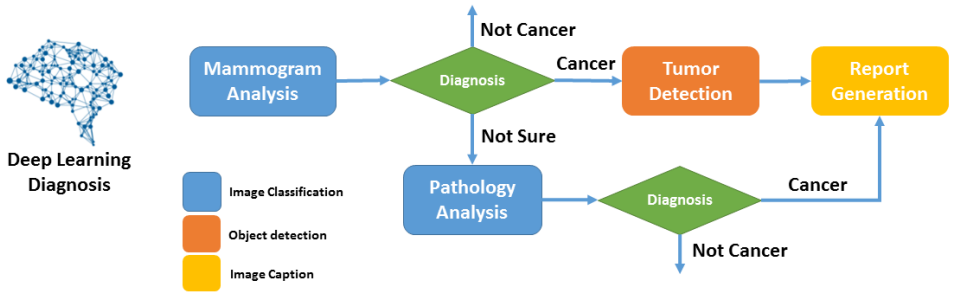

Figure 2.7 Workflow of Our Diagnosis System

1. Perform mammogram analysis first

To determine if a tumor is benign or malignant, we will first require the patient's magnification mammogram image. The deep learning program will try to make a preliminary classification: cancer, not cancer, or not sure. More detailed diagnosis should follow.

2. Detect possible tumor location if classified positive

If the program categorizes image as positive, it will further detect the exact existence of tumor. It will point out the most suspicious regions in the image for pathologists' reference.

3. Make a more confident judgment with pathology analysis

If the program cannot achieve a pre-defined certainty threshold, it will suggest a pathology analysis. As the pathology analysis can give more information, very likely the program will approach the correct inference.

4. Generate human-readable report

At the last, the program will describe its output in an understandable way. The report will indicate all its findings.

In term one, our primary objective is to build an accurate breast cancer histopathological image classification model, which is the first computer diagnosis procedure in the workflow. This is the entry point, and will be the most frequently used module in the system. Therefore, we plan to spend more effort in this part and to get a model as accurate as possible.

3 Related Works

3.1 Naïve Bayes for Breast Cancer Diagnosis

Starting from the emerging of AI, attempts were made to predict medical image classes. The work from Kowal et al. [25] used a traditional Naïve Bayes classifier for automated breast cancer diagnosis. It turned out that AI is feasible for this diagnosis since simple classifiers can also do a good job.

In their thesis, the first step was preprocessing. Their original data was not of high quality, and there were many noisy pixels in the image. They used Gaussian filter to blur the image and reduce the noise. Then they stretched the histogram to improve contrast. The second step is segmentation of nuclei, since classification of tumor requires identifying nuclei in each cell. they implemented four clustering algorithm: competitive neural network, fuzzy C-means, K-means and Gaussian mixture model. Then 42 features were extracted from each segment. The features were selected by experienced human pathologists. Then the features were feed into classifiers. They trained a Naïve Bayes classifier which was using estimated kernel densities. 500 real medical images from 50 patients formed the train dataset.

They measured the performance with n-fold cross validation method. Their accuracy rate was about 96%-100%, which indicated AI in breast cancer diagnosis was quite promising for production. It showed that their preprocessing procedure and data collecting procedure could assure accurate and objective dataset.

	KM	FCM	GMM	CNN
Patients Accuracy	100.00%	96.00%	100.00%	98.00%`
Image Accuracy	90.22%	85.78%	88.00%	89.56%

Table 3.1 Performance of Different Classifiers

3.2 SVM for Remote Breast Cancer Diagnosis

The work from George et al. [26] proposed a more advanced system for breast cancer diagnosis. They proposed a fully automatic nuclei detection and segmentation method. Then they developed the AI tumor classification system. They proposed 12 features for research on the most effective model. At last, they experimentally pushed their computer aided detection and diagnosis system to production, connecting it to a remote medical platform. This web based service was expected to provide an intelligent and convenient diagnosis for breast cancer patients.

Their first step was preprocessing. Since preprocess preprocessing is the most critical factor in image processing, they shrank the image size from 2560x1920 to 640x480. Then contrast enhancement and edge sharpening was used to manipulate the image. They used contrast limited adaptive histogram equalization to enhance the quality of the image. CLAHE worked within each tile of the image instead of the whole image, so that contrast was enhanced in each tile. The next step is cell nuclei detection. They implemented a detector combining circle detection and local maximum finder. In the images, there may exist some blood cells which were unwanted noisy markers. They used Fuzzy C-Means Clustering method to remove such cells. The noise free image was then separated into individual objects with marker-controlled watershed transform.

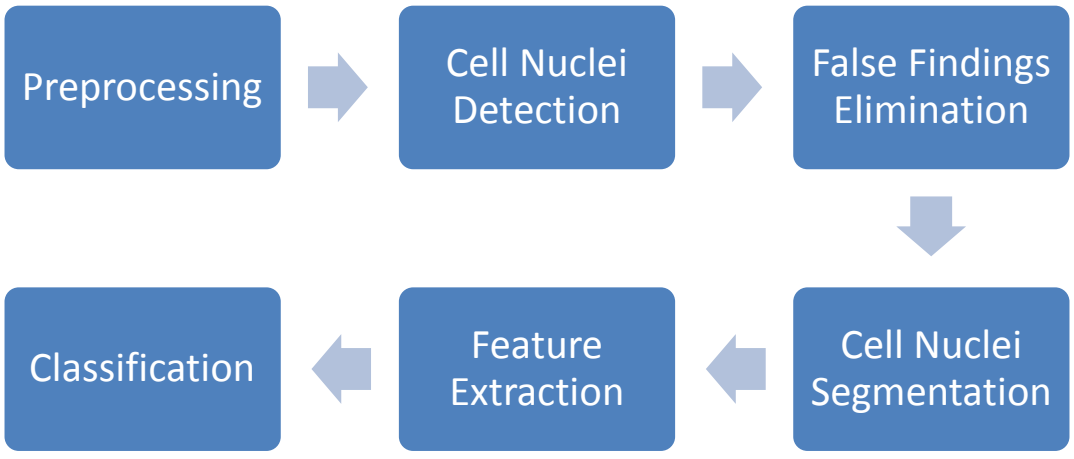

Figure 3.1 Workflow of SVM System

They used some meaningful features to classify the image. They proposed two textural features and ten shape features that could yield a good

discrimination ability. The features include boundary, smoothness, etc. Then the features were feed into SVM.

The train set and test set was generated with ten-fold cross validation method. A total of 3260 images were used in the experiment. The experiment result showed that their method was still effective for bloody images or noisy images. However, due to the extreme lack of data, their accuracy was capped at 82.6%. Some data set still did not the training goal after 200 epochs. This paper illustrated some effective ways to preprocess images, and proved that the performance converge is greatly correlated to the size of train set.

3.3 Classification of Skin Cancer with DNN

Some most recent research on medical deep learning discussed deep neural networks for classification of skin cancer. Esteva et al. [27] described a promising method to do image classification for skin cancer diagnosis.

Instead of highly standardized images generated from specialized instrument such as microscope, their classifier was mainly focused on classifying images from general purpose photography instruments like smartphone. The variety of zooming, angle and brightness brought new challenge to the task. They used data driven method to overcome this difficulty – they increased the size of dataset to 1.41 million which was impossible for standardized images. The number of images made classification more robust to the variety in images. Compared to previous work that required many preprocess, segmentation and feature extraction, they required no handmade functions in the classification. Their model directly read the original image and original pixels and perform an end to end training.

Their classification includes 2032 single diseases arranged in a tree structure. Three root nodes represented benign, malignant and non-tumor lesions. It was given in the bottom to top structure and therefore was very suitable for machine classifiers.

They utilized the GoogleNet Inception v3 CNN architecture [28], which was previously trained for 2014 ImageNet challenge, and then transferred to the skin cancer dataset with transfer learning technology. This is a deep CNN architecture which achieved 93% accuracy in the challenge. They deleted the final classification layer, and retrained the network with the skin cancer dataset, and fine tune parameters of each layer. During the train process, they shrank size of each image to 299x299 pixels so that it could fit with the input sized of the original Inception v3 network structure, and used ImageNet to pre-train the image feature learning ability of the network. This process was called transferred learning, which could result in the best result with given number of data.

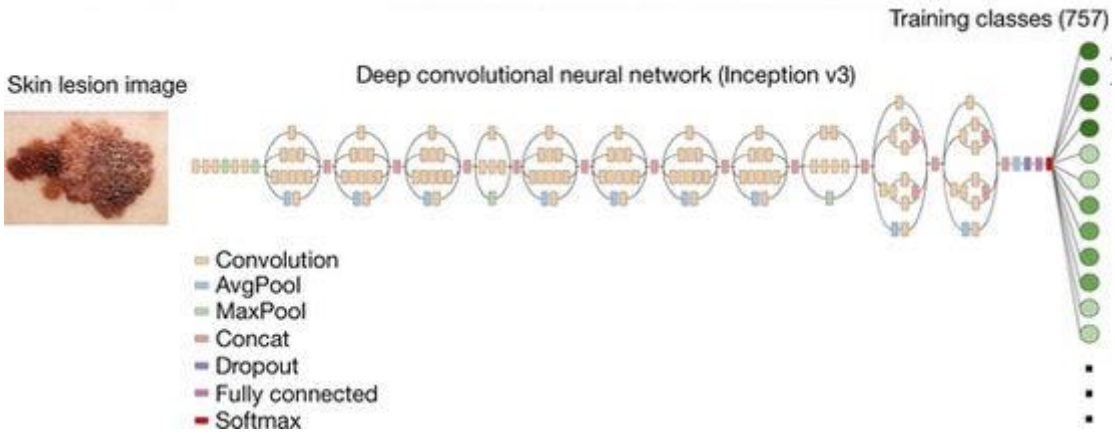

Figure 3.2 Structure of Inception v3

They trained the Convolution with back propagation algorithm. All layers in the network was assign the same global learning rate. They used Tensorflow, a deep learning framework by Google to train, validate and test their network.

They tested their network with two methods, using nine-fold cross validation. First, they used three top-level nodes for classification, which classified each image into benign, malignant or non-tumor. In this task, CNN achieved 72.1 ± 0.9 % accuracy for each patient. Two human dermatologists achieved 65.56% and 66.0% on a subset of the test set. Second, they classified images into different medical care requirements. CNN achieved 55.4 ± 1.7% while two dermatologists achieved 55.0% and

53.3%. This demonstrated the effectiveness of deep learning for cancer diagnosis. This method is mainly bounded by data; if given enough data, it can be suitable for many other image problems.

4 Technical Support and Preliminary Study

4.1 Breast Cancer Diagnosis

In this section, we will discuss about the medical background we studied for this project. Topics covered include histopathological image, pathophysiology and current diagnosis method of breast cancer.

4.1.1 Histopathological Image

Microscopic biopsy image is the standard tool for pathologists to diagnose breast cancer. Pathologists will inspect the size, shape, structure of cells and tissue and try to find some specific dangerous features in the image. Some signal used in this procedure include how each cell looks like, how each nuclei looks like and how the tissue looks like [29].

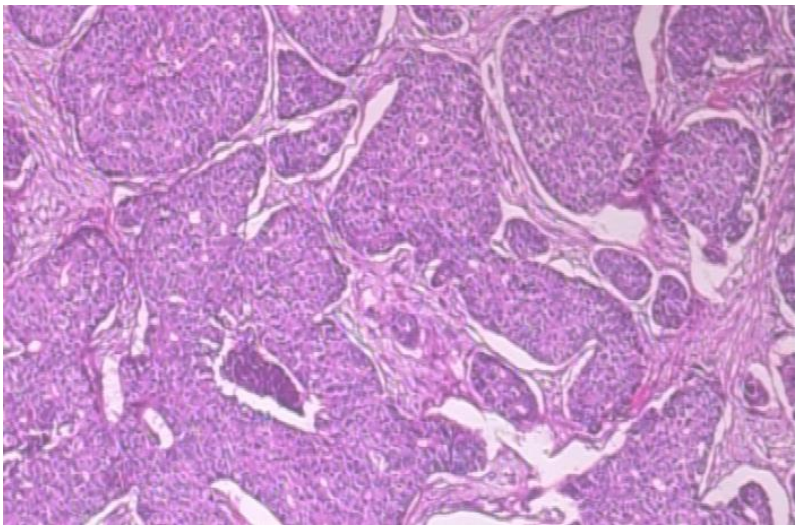

Figure 4.1 Sample of Histopathological Image

Shape and size of the cells

Observations show that cells in a piece of tissue usually do not derivate too much from the average overall size and shape. However, a cancerous cell will lose its normal appearance, being either bigger or smaller than other cells. Well-functioning cells have even shapes and structures. On the other hand, cancer cells hardly function in a meaningful way, often with their shapes uneven.

Size and shape of the cell's nucleus

Cancer cells often do not have a nucleus with normal size or shape. On the contrary to healthy nucleus, cancer nucleus is less likely to be located at the center of the cell. The cancer cell tends to have an appearance like an omelet, where the nucleus is the yolk. The nuclei of it is also bigger and darker compared with that of a normal cell.

Distribution of the cells in tissue

Besides things inside each cell, the functionality of tissue also depends on how cells are distributed and arranged. If the number of healthy cells is reduced, the overall texture and even color will also change accordingly, which leads to the shape and morphology features pathologists can directly observe from the tissue. This is more significant in diagnosis.

4.1.2 Pathophysiology

We investigated the pathophysiology explanation of breast cancer. This will help us understand features in the images, and help us develop a system more specific to our task.

Cancer is immune defense failure

The immune system normally seeks out cancer cells and cells with damaged DNA and destroys them. Breast cancer may be a result of failure of such an effective immune defense and surveillance.

Cancer involved stromal cells and epithelial cells

These are several signaling systems of growth factors and other mediators that interact between stromal cells and epithelial cells. Disrupting these may lead to breast cancer as well.

Risk factors of cancer vary

1. Age: The risk of developing breast cancer increases with age.
2. Personal history: A personal history of breast cancer is also a significant risk factor for the development of a second ipsilateral or contralateral

breast cancer.

3. Breast pathology: Proliferative breast disease is associated with an increased risk of breast cancer.
4. Family history: A woman's risk of breast cancer is increased if she has a family history of the disease.

Lifestyle contributes to cancer

1. Alcohol consumption: Alcohol consumption has been associated with increased breast cancer risk that is statistically significant.
2. Physical activity: It has been observed that frequent physical activity can lower the risk of breast cancer.
3. Obesity: Obesity, specifically in postmenopausal women, has also been shown to increase a woman's risk of breast cancer.
4. Radiation: Radiation exposure from various sources including medical treatment and nuclear explosion will increase the risk of breast cancer by a slight amount.

4.1.3 Current Diagnosis Method

We also studied the current standard diagnosis method of breast cancer. This will equip us with the knowledge about how to simplify the traditional diagnosis process.

Breast cancer screening

Breast cancer screening is defined as the medical screening process among women appear to be healthy for early symptoms of breast cancer [30]. It is proposed in the will to diagnose It is widely believed that early detection will improve patients lone-term survival rate.

Microscopic analysis of a biopsy by pathologists

If the screening result is inconclusive, the doctor may require a microscopic analysis. The doctor will sample the fluid in the lump to do a further diagnosis. This procedure involves needle aspiration. If the fluid is clear, it is highly likely that the patient is healthy; however, if there exist bloody fluid, a more detailed microscope inspect will be needed and it is possible

that the lump is affected [31].

This method is the most widely employed procedure. However, it is also laborious and time consuming. The probability of misdiagnoses is high because there can be too many variations in the process. Considering the incredible amount of data involved, it is a huge work.

4.2 Image Processing

Preprocessing is an important step in the process. The phrase "garbage in, garbage out" is particularly applicable to our project. Though the image gathering methods are often strictly controlled for our dataset (i.e. same microscope), the original data still have different attributes such as brightness, contrast and saturation. Analyzing data that has not been carefully normalized can produce misleading results. Thus, the representation and quality of input data should be assured before training.

4.2.1 Feature Detection

Feature detection is a concept in the area of computer vision and image processing. It means use computer to extract information from image and to decide if each pixel of the image belongs to a feature or not. The result of feature

Figure 4.2 Points Detected in Sample

Up till now there is no universal definition of “useful” or “accurate” features. The precise choice of features usually depends on the problem or

specific application. It is a primary computation of many computer image analysis algorithms, in other words, the start point of them. It checks each pixel to determine if a feature can be extracted from that pixel. Therefore, whether an algorithm can succeed sometimes is determined by the features it defines and uses. There are many feature detection algorithms developed to meet different kinds of requirements. Features they extract vary; their computation complexity and repeatability also differs. Some most popular shape features include perimeter, area, compactness and smoothness. Textual features such as grey scale are also used. There are no general rules for choosing features – we can only choose by experience and experiment, which adds difficulty to image classification tasks.

Fortunately, the idea of Neural Networks saves us from the work. They are designed to require little preprocessing – All the works is done automatically be the program. This ability of learning the features is the first reason why people invented Neural Networks. However, we still need some slight amends to ensure things will not go wrong.

4.2.2 Data Augmentation

There is another thing to note: data augmentation. In deep learning, to avoid the well-known overfitting problem, we usually need to feed enough data into the model. Therefore, the amount of available data sometimes is the most critical issue for deep learning. The problem is high quality data is expensive and limited. One method to overcome the shortage of data is data augmentation. We need to perform geometric transformation on the original dataset, change pixel positions of the image while keep the original features.

Figure 4.3 Demonstration of Data Augmentation

Data augmentation is very likely to improve accuracy since the model can see more samples. The exact amount depends, though. There are many ways to augment the dataset. Adding noise is an intuitive approach. More generally we have simple transformations. For sparse holes in the dataset, we can perform dimensional reduction. Several more complicated ways include combinations of rotation, translation, rescaling, flipping, shearing, and stretching [33].

4.2.3 OpenCV

Open Source Computer Vision Library (OpenCV) is an open source library dedicated to the field of machine learning and computer vision. It was built with the idea to provide a reusable common infrastructure for computer vision applications, and to encourage the use of machine learning in real products [34]. The library was originally proposed by the CPU company Intel, and was later maintained by other organizations.

Figure 4.4 Logo of OpenCV

There are more than 2500 optimized algorithm included in this library. This includes both traditional and most advanced machine learning algorithms. This brings us convenience in developing our deep Neural Network.

OpenCV support programming languages from C, C++ to Java and Python. The main focus of it is to improve computational efficiency and therefore to enable interactive applications that can respond quickly to changing

inputs. It has a backend optimized with C/C++, and can take the full advantage of multicore processors. It can also utilize hardware acceleration provided by different platform.

4.3 Deep Learning

Most importantly, we searched for the latest technology and tools in the field of deep learning. With these knowledge, we will try to build a more advanced deep learning program.

4.3.1 Convolutional Neural Network

In machine learning, convolutional neural network is a type of feed-forward neural network. It is inspired by biological processes in animal vision system [35]. Various projects have applied convolutional neural network in analyzing visual imagery. In recent years, Convolutional Neural Network has become the state-of-art in image recognition problems, beating different competitors. It has been observed from existing papers [36] [37] that CNN is feasible to do microscopic and macroscopic image classification tasks, and is possible to surpass other classifiers. It is now believed to be the first choice for image classification type tasks.

Just like other Neural Networks, CNN consists of an input layer, multiple hidden layers and an output layer. A notable feature of CNN is that it assumes inputs are pictures. In this way, it can do some more specialized optimization. In convolution layers, the neurons will only connect to a limited region of the previous layer. This reduced computation complexity, and enables CNN to make full use of the 2D structure of the input data. Therefore, compared to other deep learning architecture, CNN can often lead to better result in image or speed recognition.

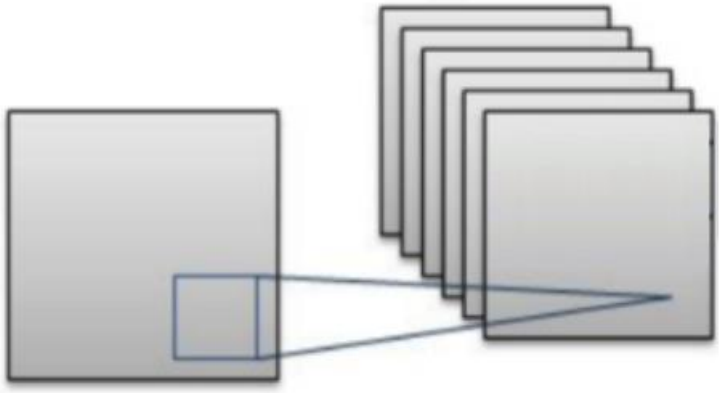

Figure 4.4 Illustration of Convolution

CNNs use less preprocessing than other image classifiers. This feature learning property reduces requirement of prior knowledge and hence human effort, making CNN an attractive architecture.

4.3.2 Residual Network

The idea of stacking up more layers is not new, but it became attracting only recently, as a result of the rapid development of Graphic Processing Units. GPUs can perform high computational intensive tasks at pretty low cost, thanks to their parallel architecture. However, as the depth of the network increase, the accuracy may not proportionally increase.

Moreover, deeper networks will face the vanishing gradient problem. The problem becomes more serious when the network is going deeper. The hidden layer near the output layer will update its weight normally, but the layers in the front of the network can only update their weights very slowly, which makes the weights almost unchanged after training. It makes the first several hidden layers merely a forward layer that do a same mapping for all inputs. The deep network is now just equivalent to a shallow network with the last several layers.

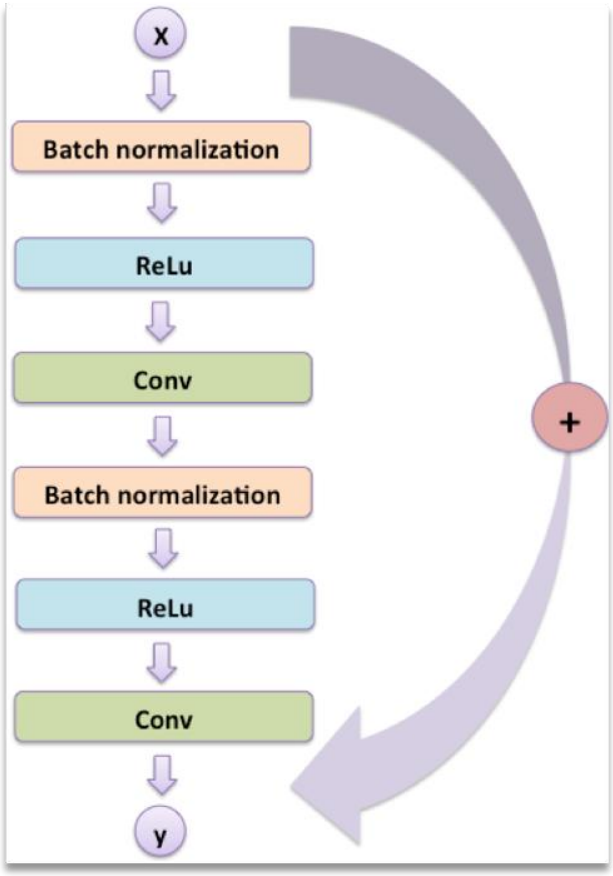

Figure 4.5 Basic Structure of Residual Network

He et al. [38] presented a residual learning framework to ease the training of networks that are substantially deeper than those used previously in 2015. They explicitly reformulated the layers as learning residual functions with reference to the layer inputs, instead of learning unreferenced functions. There is empirical evidence showing that these residual networks are easier to optimize, and can gain accuracy from considerably increased depth. On the ImageNet dataset, He evaluated residual nets and achieved 3.57% error on the ImageNet test set. This result won the 1st place on the ILSVRC 2015 classification task.

4.3.3 Tensorflow

In our project, we use Tensorflow. It is also an open source software library. By using data flow graphs, it is capable for large scale numerical computation, one of which is machine learning. Besides fast speed, it also supports various high-level APIs for machine learning programs [39].

Tensorflow supports platforms with or without GPU, from mobile, desktop to clusters. With limited overhead, Tensorflow + Python environment provides a much clearer program: we describe the data flow diagram with Python, benefiting from the conciseness of this language; then Tensorflow will execute the diagram with a C++ or CUDA backend, making full use of the computer hardware.

Figure 4.6 Logo of TensorFlow

Tensorflow introduces two new concepts: Tensor, and data flow graph. Data flow graph is a graph whose nodes are Tensors. Tensors are actually matrixes; however, they can be connected to from a data flow graph. The matrix together with the relations defines a Tensor. The word “flow” means that data will flow from one node to another, and the computation occurs in the transition. This gives us a very good simulation of CNNs: they both are graphs, and both incur computation during transitions.

We are using Tensorflow 1.3.0, which is the latest version available at the time we start to develop our project.

4.3.4 Comparing Tensorflow with Other Tools

Generally speaking, Tensorflow is more friendly to beginners than other tools like Caffe. This partly results from Google, the author of Tensorflow. Most other tools are supported by university academics, while Tensorflow is supported by a commercial company. This results in difference in

available documents, tutorials and communities. Developing with Tensorflow is generally more comfortable.

Though tools built for academics can provide a more detailed control over the model, this feature is mostly not required for implementing a model that has already been tested for many times. On the other hand, Tensorflow is more high-level, providing conciseness in development.

Developer can use Tensorboard, the bundled debug tool along with Tensorflow, to monitor real time statistics of the diagram. Considering Googles' experience in user interfaces, debugging Tensorflow models is much more convenience than debugging Caffe models.

Moreover, as Google is a commercial company, Tensorflow is designed for production usage at the very beginning. We can easily export the model trained by Tensorflow and set up a RESTful query server in a couple of lines. As our project is a medical project, we should expect users may not have much Machine Learning background. The ease in pushing experiment results to production is an advantage.

5 Method

5.1 Dataset

For our project, we are using the Breast Cancer Histopathological Image Classification (BreakHis) dataset. It is composed of 9,109 breast tumor tissue microscopic images. The researchers collected samples from 82 patients, and used different magnifying factors (40x, 100x, 200x, and 400x) to process them [40].

Class	40x	100x	200x	400x
Benign	625	644	623	588
Malignant	1370	1437	1390	1232
Total	1995	2081	2013	1820

Table 5.1 Distribution of Images

The samples are stained with hematoxylin and eosin. The author of the dataset uses breast tissue biopsy slides to generate these samples. Pathologists from the P&D lab labeled them. The breast tumor specimens were asses by Immunohistochemistry. The biopsy procedure was Surgical Open Biopsy.

An Olympus BX-50 system microscope was used to capture the images. As aforementioned, they captured image under four magnification factor, 40x, 10x 200x and 400x. The raw image was stored into the dataset without any normalization of color standardization to avoid loss of information and complexity in analysis. The images were in Portable Network Graphics (PNG) format, in 3-channel RGB, 8-bit depth.

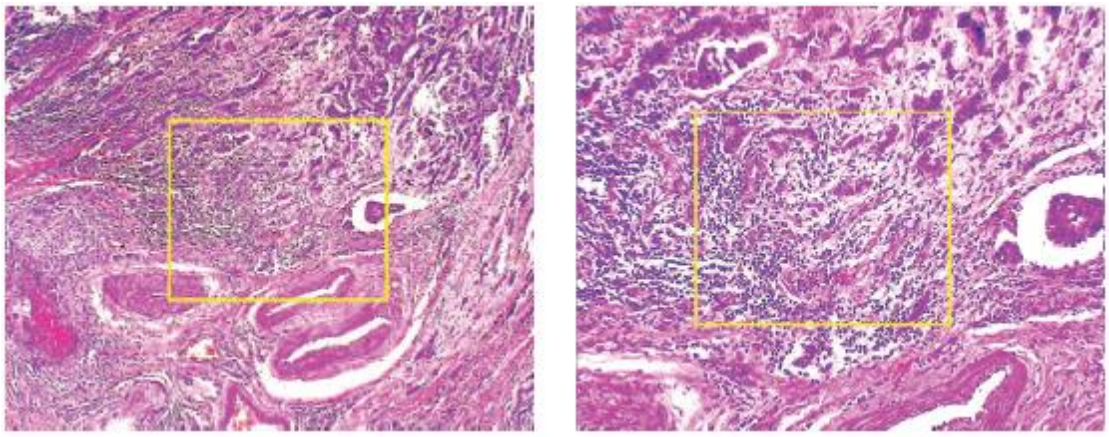

Figure 5.1 Same Tumor under Different Magnification

5.2 Preprocess

In this section, we will discuss how we manipulate the image before feeding it into the model. We proposed different methods, and would compare them in experiments.

5.2.1 Data Augmentation

Since we are training a deep learning neural network, the amount of train data is a critical problem. The size of the original dataset, 9109, is relatively small for our model, and is therefore very likely to cause overfitting. Summarizing the methods used in past works [41], we can propose multiple ways to extend the dataset systematically.

We do not propose any color standardization since all images have the same color pattern, i.e. pink or purple. This is due to the stain method applied to tissue samples. The data augmentation methods we propose include only geometric transformation. They include:

1. rotations: random with angle
2. translations: random with shift
3. flipping: true or false
4. shearing: random with angle
5. stretching: random with stretch factor between 1/1.3 and 1.3

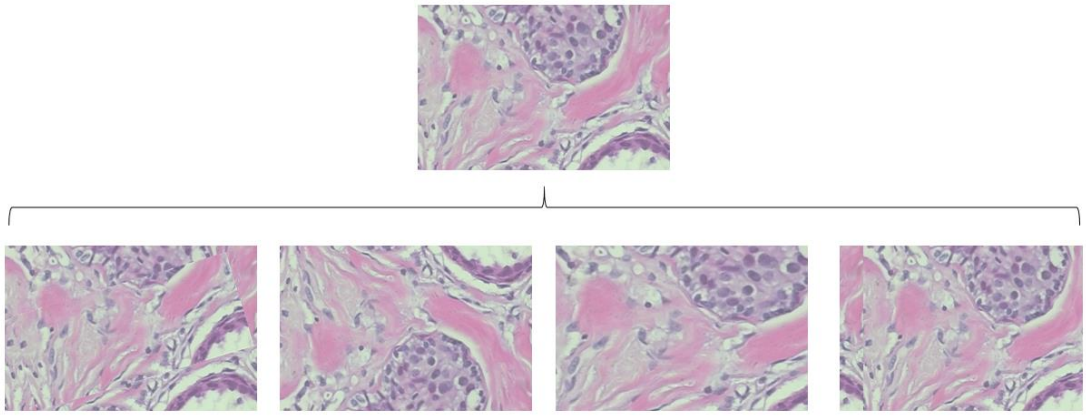

Figure 5.2 Examples of Data Augmentation

5.2.2 Sliding Window Crop

It is hard to process the high-resolution images since applying deep learning algorithms on larger image sizes will tend to make the model architecture more complicated. The model will usually have more layers, more parameters which increase the complexity dramatically. Training and tuning the model may be very costly in such case.

One way to solve this problem is sliding window crop. Set a window of size $n \times n$, slide through the image at $step = 0.5n$, and then crop [42].

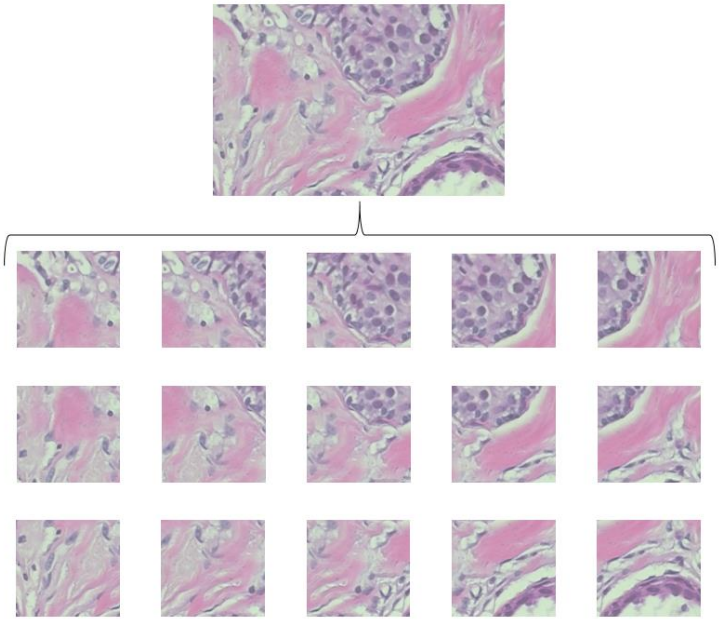

Figure 5.3 Examples of Sliding Window Crop

Overlaps between crops are deliberately designed to avoid damaging the structure information too much. The number of total crops is given by the following formula:

$$\#(\text{crop}) = 2 \times \frac{\text{IMGWIDTH}}{\text{n}} \times 2 \times \frac{\text{IMGHEIGHT}}{\text{n}}$$

5.2.3 Random Crop

Another way to solve the aforementioned oversized problem is random crop. Set a window of size $\text{n} \times \text{n}$, do random crop instead of sliding. This is similar to the previous method.

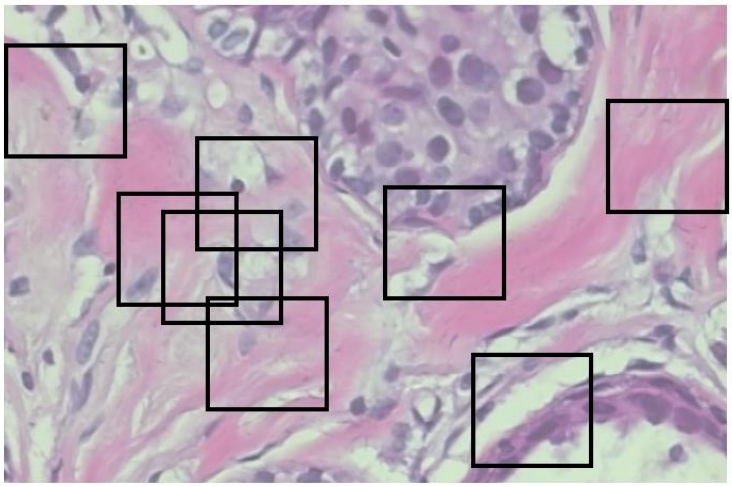

Figure 5.4 Examples of Random Crop

The number of total crops is not fixed. However, a higher number of crops will give more information. There will be no limit on how the random selector crop: it may or may not capture the most important features.

For benign samples, there will be no problem. However, for malignant samples, we cannot make sure tumor exist in every crop. Crops extracted from malignant images may actually contains no tumor and should be classifies as benign. This introduces noise in train data.

The gain, on the other hand, is we keep the size of network small. This benefits in various ways: less computation complexity, less logic complication, and most importantly, it reduces chance of overfitting by

limiting the parameters of the model to a reasonable amount.

5.2.4 Resizing

There always exists the method of simply shrinking the image. To avoid moiré after resizing, we will resample the image using pixel area relation. This is the best image interpolation method for decimation since it tends to give a clearer image. This makes the high-resolution image generation pointless, however.

5.2.5 Whitening

Whitening is the one of the standard preprocess methods for machine learning. The main idea is to remove extra information dimensions in the image. First, we represent the input dataset as

$$\{x_1, \dots, x_m\}$$

Then we computes the covariance matrix of x

$$\Sigma = \frac{1}{m}\sum_{i=1}^{m} x_i x_i^{\mathrm{T}}$$

Therefore, we can have

$$x_{\mathrm{rot}} = U^{\mathrm{T}} x$$

where U is the eigenvector of Σ.

This process maps x into a new space that eliminates the correlation between features. Then we can have

$$x_{\mathrm{PCAwhite}} = \frac{x_{\mathrm{rot}}}{\sqrt{\lambda_i}}$$

which normalizes the dataset [43].

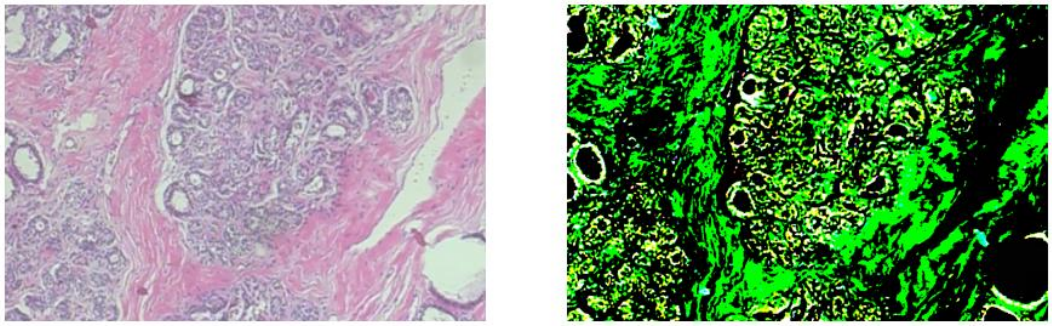

Figure 5.5 Before and After Whitening

After whitening, the new image satisfies two properties: features are less correlated, and features have the same variance. This will significantly accelerate the training process.

5.2.6 Contrast Limited AHE

Contrast-Limited Adaptive Histogram Equalization (CLAHE) can improve local contrast without damaging the image too much. Consider an image whose pixel values are limited to a specific range, it would be better to have the values distributed in all regions of the channel. This will usually improve the contrast of the image. Therefore, we need further scatter pixels clustered in the “brighter” regions.

Adaptive Histogram Equalization (AHE) will do this work. However, it sometimes will cause loss of information due to over exposing some region that is already bright. This is because the image is not perfectly limited in a small region of the channel. To solve this problem, we can use CLAHE [44]. The image is divided into tiles, and each tile can perform AHE on its own. For a tile, the brightness across this small area is more likely to be confined. In this way, the image will be clearer.

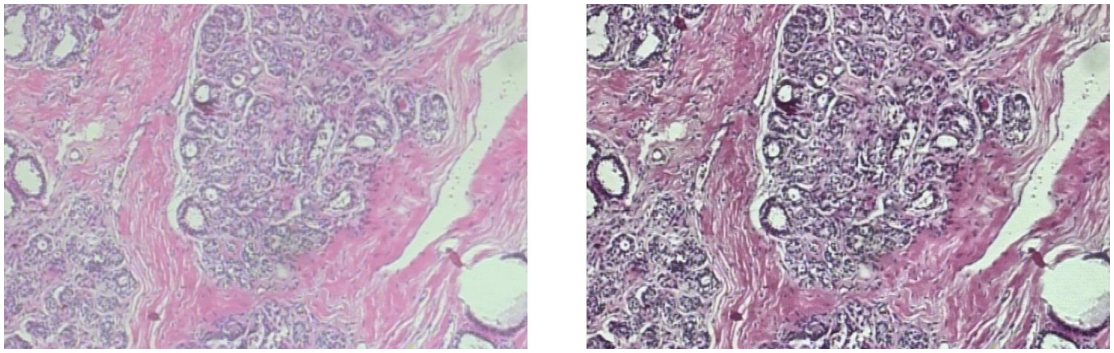

Figure 5.6 Before and After CLAHE

Generally speaking, CLAHE is more important than whitening for deep neural networks since the network can learn how to whiten images itself without manually specify it should do it.

5.3 Model Architecture

We propose deep neural network to be the framework of our deep learning program. To define a deep neural network, we will discuss about how we will construct each layer of the network in the following sections.

5.3.1 Input Layer

This is the first layer of the network. It received non-linear input data and prepare data to be fed into convolutional layers after it. Some simple transformation such as normalization can be applied in this layer. It produces the initial feature maps. In our experiment, the input is an image, and the network is parameterized according to the image width, height and depth. Since we will test multiple cropping methods, available parameter set includes multiple values, for example 700x460, 256x256, etc.

5.3.2 Convolutional Layers

The convolution layer takes data from previous layers and a group of trainable filters as input. A filter is just a neuron connected to a limited area of the previous layer. Each filter will produce a feature map in the output.

In the convolution layer, filter will do convolutional computation on local input data. The data window will keep sliding after filter finishes the local computation, until it finishes all data from the previous layer.

While the input data may have a large size, the filter will only compute the convolution on a partial data window, which is called local perception mechanism in CNN. It is a simulation of animal focusing on a specific object. Meanwhile, as the data window slides and the input data changes, the filter weight is fixed during this iteration; in other words, focusing on different area will not change the way an animal see the world. This is the weight sharing concept in CNN.

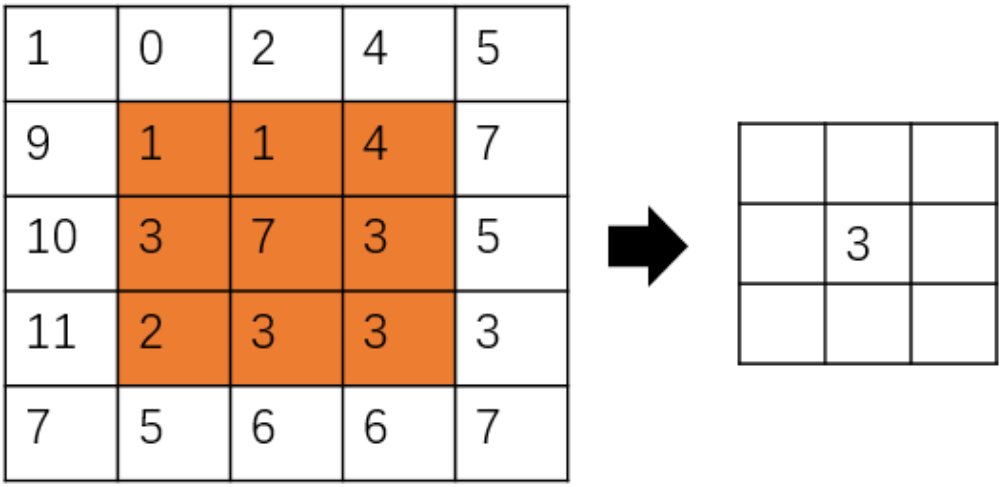

Figure 5.7 Convolution

In this procedure, we need to specify several parameters: depth, stride and zero-padding. Since we will test multiple model architecture, available parameter set also includes multiple values. We will do experiments on 3x3, 5x5, 7x7 kernels.

5.3.3 Dropout

From experience, overfitting is a common problem in deep neural network. Due to that large amount of trainable parameters in CNN, the model may simply memorize all train data without figuring out the internal regulation in the dataset, and cannot be generalized to new data, which leads to high accuracy on train set but low accuracy on test set.

Dropout refers to the method that temporarily disable some neural network unit at some certain probability during the train process of CNN. The discard is temporary, and its weight is preserved. For random gradient decline, since units are randomly disabled, the training is actually on different networks for each mini-batch. It forces one neuron to work with other randomly selected neurons, forces "free riders" to be trained equally, and hence decreased the correlation among neurons. In this way, we are actually training 2^n models for a neural network with n nodes, while keeping the number of parameters unchanged. In other words, we are training more models with the same computation complexity. This results in a visible improvement in the generalization ability of the network.

In our project, we applied a dropout layer after each convolution layer with dropout rate 0.5.

5.3.4 Residual Blocks

The representation ability of a network increases if its depth increases. For two network with same time complexity, the deeper network will perform better [38]. However, the actual classification accuracy usually does not increase for deeper networks; sometimes the accuracy even decrease.

To solve this problem, residual learning was proposed. If multiple non-linear layers can be approximated by a function, we can also represent the residual of this hidden layer as a function. Suppose a hidden layer is $H(x) - x \to F(x)$, we can intuitively have

$$H(x) = F(x) + x$$

Then we can have the residual block. The output of the residual block is the sum of the output of multiple cascade convolutional layers and the input element itself, activated by an activation function where we choose ReLU.

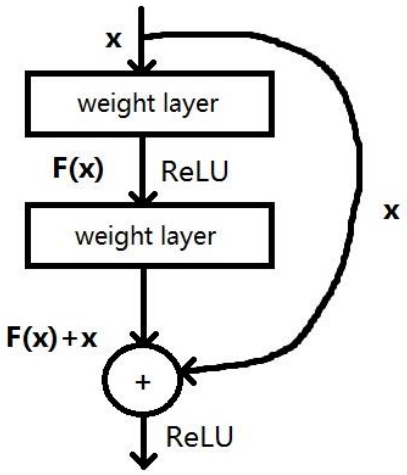

Figure 5.8 A Simplified Residual Block

The residual network has some nice features. It is thin, having the number of parameters under control. There is layered structure which can ensure the feature expression ability of the network. It can perform subsampling without pooling layers, and therefore improved the efficiency of back propagation.

For actual usage, the number of convolutional layers wrapped by a residual block may depends on scenario. For our project, we choose 5 since it is the best balance between train time and accuracy gain.

5.3.5 Pooling Layers

The pooling layers are used to perform subsampling. The size of its output will be reduces, but the depth will keep unchanged. It will reduce the amount of data and the number of parameters in the model. During the training process, it can therefore lower the computation complexity and avoid overfitting. The polling layer uses the same sliding window mechanism as convolution layers, and is defined as

$$y = \max_{\text{local window}}(x)$$

In our model, there will be a pooling layer after each convolution layer, so their activity is strictly determined by convolution layers. The filter size we used is 2x2 and the stripe size is 2, which will reduce the amount of data

by 75%. Larger sizes may be destructive to the network.

5.3.6 Activation Layers

Activation layers are introduced for adding non-liner classification ability to neural networks. Though it is logically just a function, usually we regard it as a layer. In our model, we use Rectified Linear Unit (ReLU) as the activation function. It is a commonly used one for CNN. The ReLU function is defined as

$$f(x) = \max(0, x)$$

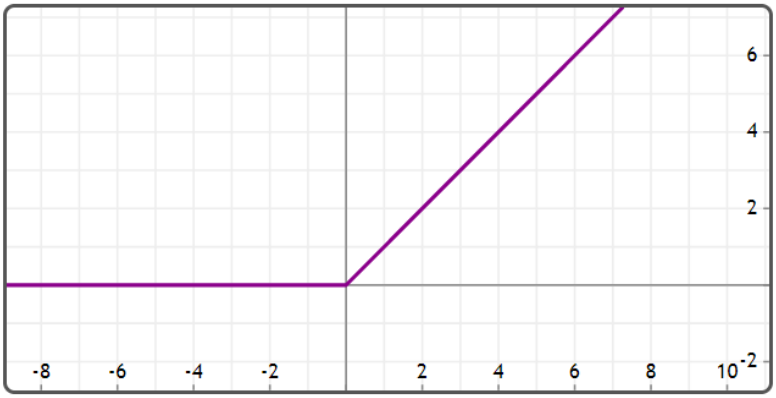

Figure 5.9 ReLU

As shown in the graph, the activation function ReLU that we used is just a threshold at zero. It is proved to be a better simulation of animal brains [45]. For particle usage, it simplifies the computation required.

5.3.7 Fully Connected Layer

The fully connected layer has connection to all neutrons of the previous layer. We have only one fully connected layer. It is used at the end of the network to produce final prediction results.

5.4 Aggregation

As aforementioned, some images are sliced into patches; we need to aggregate patches to get the classification for the whole image. For patient

level diagnosis, we also need to draw an overall conclusion from all histopathological slides. We propose different aggregation rules inspired by Kittler [46] here, and will test their performance for both two tasks.

5.4.1 Sum

This aggregation rule assumes that a posteriori probability will not deviate too much from the prior knowledge. It takes into consider the ratio of benign samples and malignant samples among all patients. In such case we can assume that the posterior satisfies

$$P(w_k|x_i) = P(w_k)(1+\delta), \delta \ll 1$$

Therefore

$$P(w_k)\prod(1+\delta_k) = P(w_k) + P(w_k)\sum \delta_k$$

By applying the Bayes' theorem, we have

$$P = (1-R)P(w_k) + \sum(w_k|x_i)$$

We will assign

$$\text{Prediction} = \text{argmax}[(1-R)P(w_k) + \sum P(w_k|x_i)]$$

5.4.2 Plurality Vote

This is a simple rule that reports the majority of all samples. However, this method is not likely to perform well, since an image should be classified as malignant once there is a tumor, no matter if tumor cells takes up the majority space of the image. We use this as the baseline for assessment. We will assign

$$\text{Prediction} = \text{argmax}(\sum \Delta_i)$$

5.4.3 Average

This rule can be view as a more advanced voting. It computes the average of predictions for each class over all samples; then it performs maximum

likelihood estimate. We will assign

$$\text{Prediction} = \text{argmax}(\frac{1}{\text{R}}\sum P(w_k|x_i))$$

Thus, this rule assigns classes in a more reasonable manner, since an outliner tumor can still affect the average and therefore affect the final classification result. By summing up all predictions, it balances between popular opinion and individual judgement.

5.4.4 Exist

This rule is another extreme, in contrast to plurality vote. Once if a tumor is detected, the final prediction will be malignant. This rule will impose more false positive, but may be more useful in real applications since people are more tolerant to false positive than false negative. We will assign

$$\text{Predition} = \begin{cases} \text{malignant}, & \sum \Delta_i > 0 \\ \text{benign}, & \sum \Delta_i = 0 \end{cases}$$

5.4.5 Exist-n

This rule is a variant of the exist rule. It adds a threshold of n to the prediction, in other words, the final prediction will be malignant if and only if three samples report tumor at the same time. We will assign

$$\text{Predition} = \begin{cases} \text{malignant}, & \sum \Delta_i \geq \text{n} \\ \text{benign}, & \sum \Delta_i < \text{n} \end{cases}$$

5.5 Workflow

To develop a more accurate model, we have the following development workflow cycle. The cycle includes five elements: design, implement, train,

validate and test.

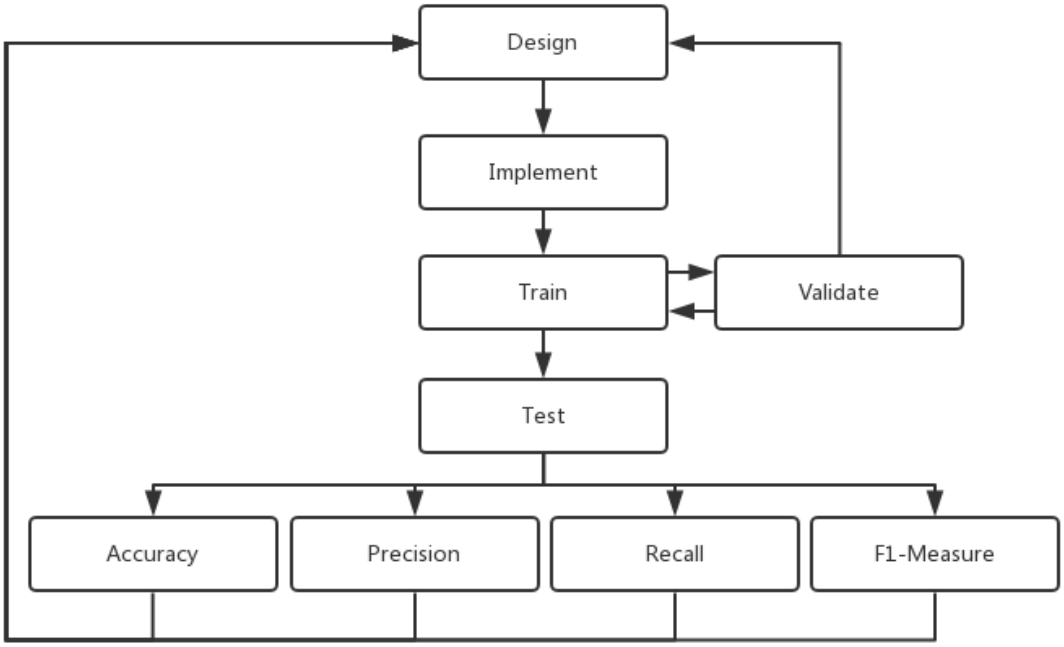

Figure 5.10 Development Workflow

After we implement a model, we will train it. Validation will be performed occasionally. If the validation result is not satisfying, we will cut off the training and attempt to find out the reason. After the train accuracy converges, we will do a thorough test of the model and compute some quantitative measurement to determine if our design works well. We will try to analyze the factors that affect the performance, and perform incremental modifications accordingly.

6 Implementation

Applying and combining methodologies we mentioned above, we have successfully implemented a ResNet model, and trained it with preprocessed BreakHis dataset, which has also been introduced before. In this part, we will divide our implementation to several parts according to code logic and introduce details of each part one by one. Also, we will explain and show our assigned parameters in this section.

6.1 Data Loader and Preprocess

This part is about the implementation of loading data and preprocessing images to fit them into ResNet model.

6.1.1 Data Loader

There are kinds of diseases such as mucinous carcinoma and adenosis in original dataset. Each disease is divided into two classes, benign and malignant, and has a file to record paths and image numbers of it.

```
SOB/mucinous_carcinoma/SOB_M_MC_14-18842/200X : 16
SOB/mucinous_carcinoma/SOB_M_MC_14-18842/400X : 9
SOB/mucinous_carcinoma/SOB_M_MC_14-18842/100X : 22
SOB/mucinous_carcinoma/SOB_M_MC_14-18842/40X : 15
SOB/mucinous_carcinoma/SOB_M_MC_14-13418DE/200X : 14
SOB/mucinous_carcinoma/SOB_M_MC_14-13418DE/400X : 11
SOB/mucinous_carcinoma/SOB_M_MC_14-13418DE/100X : 15
SOB/mucinous_carcinoma/SOB_M_MC_14-13418DE/40X : 15
SOB/mucinous_carcinoma/SOB_M_MC_14-18842D/200X : 16
```

Figure 6.1 Example of One Record File

To read all files recording the path of each disease, we use regex in python and store all image paths in a list ***shuffled_walk*** :

```
  shuffled_walk = []
   regex =
re.compile('(?:{data_set_dir}).(\w+).SOB.(\w+).([\w-]+).({magn})X'.format(
       data_set_dir=re.escape(data_set_dir),
       magn=re.escape(str(FLAGS.magn))
   ))
   for dirname, _, filenames in os.walk(data_set_dir):
       if regex.match(dirname):
           tumor_class, tumor_type, slide_id = regex.match(dirname).group(1, 2, 3)
           shuffled_walk.append((dirname, filenames, tumor_class, tumor_type,
slide_id))
   print (shuffled_walk)
```

Figure 6.2 Read All Files

To divide data into train dataset and test dataset, and keep them unchanged on accuracy test of different model, we simply use remainder of ***shuffled_walk*** 's index to divide the data:

```
i1 = [ i for i in range(len(shuffled_walk)) if i%4 == 0]
i2 = [ i for i in range(len(shuffled_walk)) if i%4 != 0]
index = i1 + i2
tmp = shuffled_walk
shuffled_walk = []
for i in range(len(tmp)):
    shuffled_walk.append(tmp[index[i]])
```

Figure 6.3 Divide Data into Test and Train Dataset

6.1.2 Image Segmentation

After storing image paths, we need to do image segmentation to fit image with proper size into our DNN model. As mentioned above, we use three kinds of strategies to do image segmentation: sliding window crop,

random crop and resizing, which have been introduced in section 5.2 Preprocess. The implementation and explanation of arguments and outputs are as follow:

Args:

Image: 3-d numpy array. The raw image to be segmented.

sub_slides: list. The list of segmented batch.

Returns:

A new list of segmented batch containing the segmentation result of input image.

```
def resizing(image,sub_slides):
    image_shape = np.shape(image)
    indexes = np.random.choice(image_shape[1] - image_shape[0], 50)
    for i in indexes:
        sub_slides.append(image[:,i:i+IMG_HEIGHT])
    return sub_slides
```

Figure 6.4 Resizing

```
def sliding_window_crop(image, sub_slides):
    image_shape = np.shape(image)
    col_step = int(IMG_WIDTH / 2 - (IMG_WIDTH - image_shape[0] %
IMG_WIDTH) / (image_shape[0] / IMG_WIDTH * 2))
    row_step = int(IMG_HEIGHT / 2 - (IMG_HEIGHT - image_shape[1] %
IMG_HEIGHT) / (image_shape[1] / IMG_HEIGHT * 2))
    for col in range(0, image_shape[0] - IMG_WIDTH + 1, col_step):
        for row in range(0, image_shape[1] - IMG_HEIGHT + 1,
row_step):
            sub_slides.append(np.array(image[col:col + IMG_WIDTH,
row:row +IMG_HEIGHT]))
    return sub_slides
```

Figure 6.5 Implementation of Sliding Window Crop

```
def random_crop(image,sub_slides):
    image_shape = np.shape(image)
    x = np.random.choice(image_shape[0]-IMG_HEIGHT,100)
```

```
    y = np.random.choice(image_shape[1]-IMG_WIDTH,100)
    for i in range(100):
        sub_slides.append(image[x[i]:x[i]+IMG_HEIGHT,y[i]:y[i]+IMG_W
IDTH])
    return sub_slides
```

Figure 6.6 Implementation of Random Crop

6.1.3 Image Preprocess

After slicing images into patches, we implemented different preprocess methods to test whether is suitable for histopathological image, which has been introduced before (section 5.2). The implementation and explanation of arguments and outputs are as follow:

Args:

Image: 3-d numpy array. The raw image to be preprocesed.

Returns:

Image: 3-d numpy array. A new preprocessed image.

```
def whitening_image(image_np):
    for i in range(np.shape(image_np)[0]):
        mean = np.mean(image_np[i])
        # Use adjusted standard deviation here, in case the std ==
0.
        std = np.max([np.std(image_np[i]), 1.0 /
                      np.sqrt(IMG_HEIGHT * IMG_WIDTH * IMG_DEPTH)])
        image_np[i] = (image_np[i] - mean) / std
    return image_np
```

Figure 6.7 The implementation of whitening

```
def subtract_gaussian_smooth_image_and_CLAHE(image_np):
    for i in range(np.shape(image_np)[0]):
```

```
        blur = cv2.GaussianBlur(image_np[i], (GAUSSIAN_KERNEL_SIZE,
GAUSSIAN_KERNEL_SIZE), 0)
        clahe_input = cv2.cvtColor(image_np[i] - blur,
cv2.COLOR_BGR2YUV)
        clahe = cv2.createCLAHE(clipLimit=2.0, tileGridSize=(8, 8))
        clahe_input[:, :, 0] = clahe.apply(clahe_input[:, :, 0])
        image_np[i] = cv2.cvtColor(clahe_input, cv2.COLOR_YUV2BGR)
    return image_np
```

Figure 6.8 One Version of CLAHE Implementation

```
def CLAHE_image(image_np):
    for i in range(np.shape(image_np)[0]):
        clahe_input = cv2.cvtColor(image_np[i], cv2.COLOR_BGR2YUV)
        clahe = cv2.createCLAHE(clipLimit=2.0, tileGridSize=(8, 8))
        clahe_input[:, :, 0] = clahe.apply(clahe_input[:, :, 0])
        image_np[i] = cv2.cvtColor(clahe_input, cv2.COLOR_YUV2BGR)
    return image_np
```

Figure 6.9 Another Version of CLAHE Implementation

```
def past_pre(image_np):
    mean = np.mean(image_np,axis=0)
```

Figure 6.10 Method in Past Paper [47]

6.2 Model

Our model is not implemented in a single inference function, but we implement functions for different usage. The parameters and outputs of each function are explained and shown in section 11, Appendix.

6.3 Train and Validation

As usually used in DNN, a model is trained by firstly feeding it input and generates the output (prediction) for comparison with the label of input.

This kind of comparison is done by calculating the loss. We used cross entropy to represent loss function [48].

```
  def loss(self, logits, labels):
       labels = tf.cast(labels, tf.int64)
       cross_entropy =
tf.nn.sparse_softmax_cross_entropy_with_logits(logits=logits,

   labels=labels, name='cross_entropy_per_example')
       cross_entropy_mean = tf.reduce_mean(cross_entropy,
name='cross_entropy')
       return cross_entropy_mean
```

Figure 6.11 Loss Function Implementation

6.4 Hyper-parameters

In this section, we will briefly explain the parameters related to research results we used and its assigned value.

learning rate: 0.001, initial leaning rate.

learning rate decay factor: 0.5, how much to decay the learning rate each time.

decay_step_0: 500, the first step to decay the learning rate.

decay_step_1: 2000, the second step to decay the learning rate.

weight decay: 0.0002, weight decay for L2 regularization.

train batch size: 64

dropout proportion: 0.5

train steps: 3000

regularizer: L2 regularizer, a process of introducing additional information to reduce overfitting.

7 Results

In the previous part, we introduced kinds of methods to do image preprocess, image segmentation and model construction. In this part, we will compare different methods and parameters together and compete with the past paper result using the same dataset to see whether our model is optimized enough.

Following the experimental protocol proposed in [40], we used cross-validation method [49] to do evaluation, the dataset was split so that patients used to build the training set (75% patients) are not used for the testing set (25% patients) to guarantee that our model can generalize to those patients not in the dataset, the results presented in this work are the average of four trials with the selected results after converging and a suitable early stop.

Training protocol used here is the purely supervised type, the Stochastic Gradient Descent (SGD) method [50], with backpropagation to compute gradients was used to update the network's parameters. All fixed hyper-parameters of training are given in the Implementation section.

The ResNet model were trained on a NVIDIA Tesla K40m GPU [53] using the Tensorflow framework [39]. Training took about 5 hours for the 256×256 input size and 10 hours for the 512×512, which is corresponding to a much more complex training set.

When we discuss the results of medical images, there are three ways to report the results in our report: batch level, image level and patient level.

Batch level can be understood by batch-wise, the unit is simply each input we fit into the neuron network. The recognition error at the image level can be calculated by:

$$\text{Image Recognition Accuracy} = \frac{N_{correct}}{N_{all}}$$

Where $N_{correct}$ is the number of cancer images which is correctly

classified, and N_{all} is the number of cancer images in the test dataset.

Patient level is a little different, each patient score is defined as:

$$\text{Patient Score} = \frac{N_{correct-in-p}}{N_p}$$

Where $N_{correct-in-p}$ is the number of cancer images of Patient P which is correctly classified, N_p is the number of cancer images of Patient P. Then the global patient error is calculated by:

$$\text{Patient Error} = 1 - \frac{\sum \text{Patient Score}}{\text{Total Number of Patients}}$$

Besides basic error results, we also calculated confusion matrix, precision, recall and F1 score [54] on either/both image level or/and patient level. Precision, recall and F1 score are defined as:

$$\text{Precision} = \frac{\text{Number of images labeled and predicted as malignant}}{\text{Number of images labeled as malignant}}$$

$$\text{Recall} = \frac{\text{Number of images labeled and predicted as malignant}}{\text{Number of images predicted as malignant}}$$

$$\text{F1 score} = \frac{2 \times \text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}}$$

Also, we use Area under the curve (AUC) [55] to measure the performance of different models, the AUC of a classifier is equal to the probability that the classifier will rank a randomly chosen positive example higher than a randomly chosen negative example, i.e.

$$AUC = P(score(x+) > score(x-))$$

Because there are millions of parameters and hundreds of hyper-parameters, therefore which parameters need to be tuned among the test should be considered carefully. Through our study on both medical and deep learning field, we selected three major hyper-parameters (directions) to do our test: Preprocess method, model architecture and image segmentation method.

For each hyper-parameter, we tested kinds of values or situations based on our guess and motivation, so each block below will contain several sub-

blocks to explain each guess and its corresponding results in detail.

7.1 Results of Different Image Preprocess Methods

Preprocess is one of the most important part in image classification, especially in histopathological image classification. According to the previous section, we have introduced different kinds of preprocess method and showed the code, in this part, we will test different preprocess method by keeping other parameters same.

Typically, the model architecture of all cases in this part is normal model architecture (Figure 7.1, left) and all inputs are segmented by different functions with size 256×256.

Figure 5.4 and 5.5 show the preprocess results of one given image to offer reader an intuitive feeling. Table 11.2.1-11.2.7 in the section 11, appendix, report the results of different preprocess methods in both batch level and image level in detail respectively, while table 7.1 is a rough comparison among different preprocess methods.

From the table below, we can find that different preprocess method has a huge influence on the results, typically, CLAHE shows a best performance on the higher magnification, where it shows that it is able to achieve an accuracy of about 5% better than the results of raw input. However, CLAHE won't work when the magnification factor is 40× while whiten operation can help model to overcome this problem.

magnification	preprocess method	image level			batch level	
		best aggregation method	accuracy (%)	F1 score (%)	accuracy (%)	AUC (%)
40×	raw	vote	81.95	86.85	80.03	80.68
	Gaussian, CLAHE	exist	68.42	80.37	65.09	68.89
	CLAHE, whiten	vote	**87.03**	**91.05**	**86.17**	**82.80**
	CLAHE	exist	81.20	89.59	82.93	82.19
	whiten, CLAHE	average	86.28	90.48	85.84	80.97
	whiten	vote	86.64	90.96	85.82	78.65
	demean	vote/average	79.51	85.64	79.42	82.41
100×	raw	exist3	78.64	85.58	79.09	79.42
	Gaussian, CLAHE	vote	69.12	80.41	69.28	70.39
	CLAHE, whiten	exist3	81.69	87.50	81.44	79.42
	CLAHE	exist3	**84.74**	**89.39**	**83.37**	76.98
	whiten, CLAHE	vote	82.23	87.76	81.42	**82.23**
	whiten	vote	83.12	88.25	82.32	82.19
	demean	exist3	79.89	86.10	79.01	81.54
200×	raw	vote/average	**88.87**	92.13	87.74	**88.36**
	Gaussian, CLAHE	average	77.19	83.83	75.90	81.52
	CLAHE, whiten	vote	85.77	90.15	84.96	85.41
	CLAHE	vote	**88.87**	**92.87**	**88.33**	85.02
	whiten, CLAHE	vote/average	85.22	89.79	84.65	87.63
	whiten	average	85.22	89.73	84.31	86.22
	demean	vote/average	84.67	89.45	83.91	82.62
400×	raw	exist	82.99	88.22	81.09	**85.73**
	Gaussian, CLAHE	vote	80.37	85.64	78.26	82.15
	CLAHE, whiten	exist3	80.56	86.73	80.15	82.05
	CLAHE	exist3	**86.73**	**90.62**	**86.15**	82.61
	whiten, CLAHE	vote	82.80	87.99	81.75	83.38
	whiten	vote/average	81.31	87.01	80.49	83.11
	demean	exist	84.67	89.24	82.96	82.97

Table 7.1. Overall results using different preprocess methods

7.2 Results of Different Model Architecture

Model architecture is also one of the features we selected to test the result and it is usually the most critical part in DNN. Previous section has introduced the basic structure of residual block, in this part, we will

evaluate the results of slightly different model architectures, which are all based on residual blocks.

Tested architectures follow the form in Figure 7.1 and 7.2; the detail can be found in Table 7.2. These networks' inputs are segmented by Resizing (section 5.2.4) and preprocessed by method CLAHE (section 5.2.6) with size 256×256.

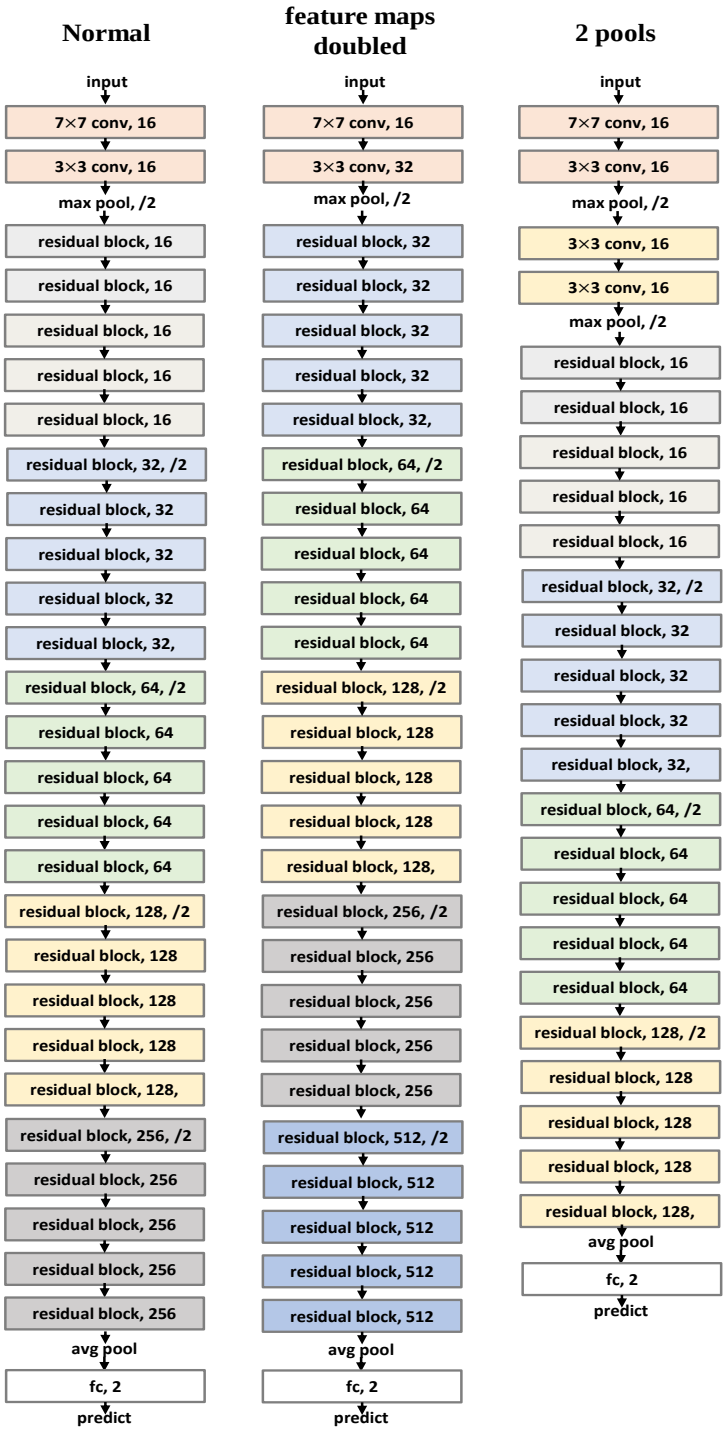

Figure 7.1 Normal, Feature Map Doubled and 2 Pools Architecture

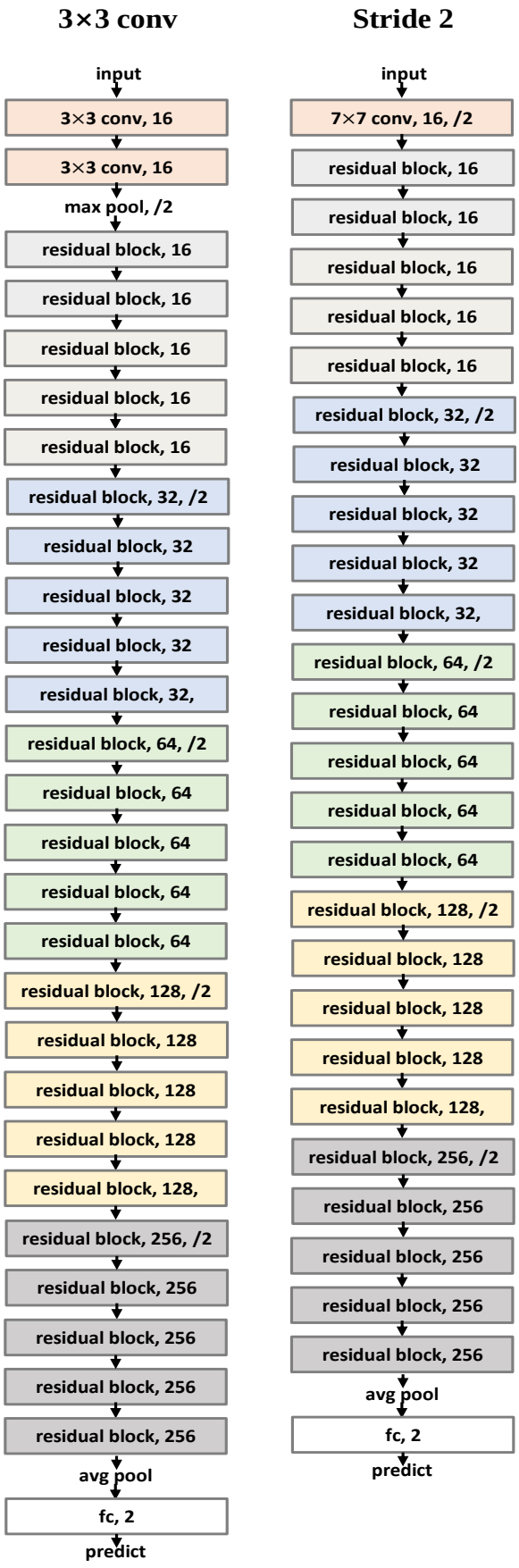

Figure 7.2 3×3 Conv and Stride 2 Architecture

Layer name	Output size	Normal	3×3 conv	Stride conv	Block number changed	Feature map doubled	2 pools
conv00_x	256×256	7×7, 16	3×3, 16	7×7, 16, stride 2	7×7, 16		
	256×256	3×3, 16		NA	3×3, 16	3×3, 32	3×3, 16
	128×128	2×2 max pool, stride 2			2×2 max pool, stride 2		
conv01_x	128×128	NA					3×3, 16
	128×128						3×3, 16
	64×64						2×2 max pool, stride 2
conv1_x	128×128	$\begin{pmatrix} 3\times3 & 16 \\ 3\times3 & 16 \end{pmatrix}\times5$			$\begin{pmatrix} 3\times3 & 16 \\ 3\times3 & 16 \end{pmatrix}\times4$	$\begin{pmatrix} 3\times3 & 32 \\ 3\times3 & 32 \end{pmatrix}\times5$	NA
conv2_x	64×64	$\begin{pmatrix} 3\times3 & 32 \\ 3\times3 & 32 \end{pmatrix}\times5$			$\begin{pmatrix} 3\times3 & 32 \\ 3\times3 & 32 \end{pmatrix}\times4$	$\begin{pmatrix} 3\times3 & 64 \\ 3\times3 & 64 \end{pmatrix}\times5$	$\begin{pmatrix} 3\times3 & 16 \\ 3\times3 & 16 \end{pmatrix}\times5$
conv3_x	32×32	$\begin{pmatrix} 3\times3 & 64 \\ 3\times3 & 64 \end{pmatrix}\times5$			$\begin{pmatrix} 3\times3 & 64 \\ 3\times3 & 64 \end{pmatrix}\times5$	$\begin{pmatrix} 3\times3 & 128 \\ 3\times3 & 128 \end{pmatrix}\times5$	$\begin{pmatrix} 3\times3 & 32 \\ 3\times3 & 32 \end{pmatrix}\times5$
conv4_x	16×16	$\begin{pmatrix} 3\times3 & 128 \\ 3\times3 & 128 \end{pmatrix}\times5$			$\begin{pmatrix} 3\times3 & 128 \\ 3\times3 & 128 \end{pmatrix}\times7$	$\begin{pmatrix} 3\times3 & 256 \\ 3\times3 & 256 \end{pmatrix}\times5$	$\begin{pmatrix} 3\times3 & 64 \\ 3\times3 & 64 \end{pmatrix}\times5$
conv5_x	8×8	$\begin{pmatrix} 3\times3 & 256 \\ 3\times3 & 256 \end{pmatrix}\times5$			$\begin{pmatrix} 3\times3 & 256 \\ 3\times3 & 256 \end{pmatrix}\times5$	$\begin{pmatrix} 3\times3 & 512 \\ 3\times3 & 512 \end{pmatrix}\times5$	$\begin{pmatrix} 3\times3 & 128 \\ 3\times3 & 128 \end{pmatrix}\times5$
	2×1	average pool, fc, softmax					

Table 7.2. Detailed architectures of evaluated models, building blocks are shown in brackets (see also Figure. 4.5) with the numbers of blocks stacked. Down-sampling is performed by conv1_1, conv2_1, conv3_1, conv4_1, and conv5_1 with a stride of 2.

7.2.1 Normal Model Architecture

Our normal model architecture is shown on the figure above (Figure. 7.1 left). The first layer is 7×7 convolution, followed by a 3×3 convolution and a 2×2 max pool with stride 2, to reduce the input size of following residual network. Then we use a stack of $NumOfSize \times NumOfBlocks$ residual blocks with 3×3 convolution on the feature maps of sizes {128,64,32,16,8} respectively. $NumOfSize$ represents the number of sizes, and the value is 5 in this model. $NumOfBlocks$ is one of the hyper-parameters we can set, the value is also 5 in this model. Table 11.2.4 in the Appendix section shows the results of normal model and the analysis of the result will be discussed later while we will focus on the comparison with this base model in this section.

7.2.2 First Convolution with Kernel Size 3×3

In our medical research, our current goal is to classify the tumor. However, there are four kinds of magnification factors in our dataset, which means that tumor in different images may have different sizes, for example, tumor in 40× image is much smaller than 400× image.

According to Table 11.2.4 we have mentioned last part, the model gained a nice accuracy on both 400× and 200× images, but is not so exciting on smaller magnification factors such as 40×, we doubted that it was due to the small tumor so that 7×7 first convolution is too big to catch the local feature of small tumor. Therefore, we tried to reduce the 7×7 to 3×3 of the first convolution's kernel, and detected its performance on small magnification factor.

(Figure. 7.2 left) shows the basic structure of this model and Table 7.3 indicates the performance of this model, surprisingly, we found that 3×3 convolution model gains a better batch level accuracy on bigger magnification factor and a better AUC on all factors. We analyze the result and think the reason may be that local feature of tumor is always smaller than 7×7, no matter what the magnification factor is. Therefore, smaller first convolution layer's kernel size can gain a better result in tumor classification tasks, which is different from the ResNet used in normal image classification problems [38].

magnification	**Image level**					**Batch level**			
	aggregation method	**accuracy (%)**	**precision (%)**	**recall (%)**	**F1 score (%)**	**accuracy (%)**	**confusion matrix**		
40×	sum	81.77	89.83	**83.91**	86.76	81.91	**AUC(%)**	83.03	
	vote	**82.33**	96.61	80.66	87.92				predict
	average	**82.33**	96.61	80.66	87.92				malignant benign
	exist	81.95	**99.44**	78.92	88.00		actual	malignant	17013 687
	exist3	82.14	98.87	79.37	**88.05**			benign	4126 4774
100×	sum	81.15	88.74	**83.46**	86.02	83.57	**AUC(%)**	84.91	
	vote	83.84	94.51	83.09	88.43				predict
	average	83.84	94.51	83.09	88.43				malignant benign
	exist	**84.02**	**98.90**	80.90	**89.00**		actual	malignant	17147 1087
	exist3	**84.02**	97.52	81.61	88.86			benign	3496 6170
200×	sum	86.86	95.89	86.00	90.67	88.31	**AUC(%)**	89.78	
	vote	88.87	99.18	86.19	92.23				predict
	average	**89.05**	99.18	**86.40**	**92.34**				malignant benign
	exist	86.86	**100.0**	83.52	91.02		actual	malignant	17996 254
	exist3	87.96	**100.0**	84.69	91.71			benign	2949 6201
400×	sum	86.54	96.22	**84.87**	90.19	86.82	**AUC(%)**	89.84	
	vote	87.10	98.84	83.95	90.79				predict
	average	**87.29**	99.13	83.99	**90.93**				malignant benign
	exist	86.36	**100.0**	82.49	90.41		actual	malignant	16989 244
	exist3	86.36	99.71	82.65	90.38			benign	3287 6280

Table 7.3. The results of model with first convolution layer's kernel size (3×3)

7.2.3 First Convolution with Stride 2

We perform down-sampling by pool layers in normal model, in this model,

we changed the stride of the first convolution from 1 to 2 and discarded the pool layer before residual blocks, which is similar with the model architecture in [38], this work is motivated to evaluate the influence of convolutional layer and pool layer we added before residual blocks.

The model structure is briefly shown in Fig 7.4, right, and the detailed results is in Table 7.4. The main discovery from Table 7.4 is that no matter what the magnification is, the best aggregation method is always vote or average, which are the most valid methods. At the same time, stride 2 shows a almost wonderful results comparing with normal model, which means that stride is usually better than pool layer when doing down-sampling.

magnification	**Image level**					**Batch level**			
	aggregation method	**accuracy (%)**	**precision (%)**	**recall (%)**	**F1 score (%)**	**accuracy (%)**	**confusion matrix**		
40×	sum	85.15	91.53	**86.86**	89.13	85.16	**AUC(%)**	86.97	
	vote	**86.65**	96.61	85.29	**90.60**				predict
	average	86.28	96.33	85.04	90.33				malignant benign
	exist	81.01	**99.15**	78.17	87.42		actual	malignant	16932 768
	exist3	83.08	**99.15**	80.14	88.63			benign	3179 5721
100×	sum	82.22	90.38	83.72	86.92	84.66	**AUC(%)**	79.13	
	vote	85.82	97.25	83.69	89.96				predict
	average	**86.00**	97.53	**83.73**	**90.10**				malignant benign
	exist	84.02	**99.45**	80.62	89.05		actual	malignant	17494 746
	exist3	84.92	**99.45**	81.53	89.60			benign	3534 6126
200×	sum	87.41	95.89	**86.63**	91.03	87.19	**AUC(%)**	85.07	
	vote	87.59	98.36	85.27	91.35				predict
	average	**87.77**	98.36	85.48	**91.46**				malignant benign
	exist	85.40	**100.0**	82.02	90.12		actual	malignant	17857 393
	exist3	86.50	**100.0**	83.14	90.80			benign	3116 6034
400×	sum	85.05	92.73	**85.29**	88.86	85.68	**AUC(%)**	87.09	
	vote	85.98	96.80	83.88	89.88				predict
	average	**86.36**	97.09	84.13	**90.15**				malignant benign
	exist	84.67	**98.84**	81.34	89.24		actual	malignant	16563 672
	exist3	85.23	**98.84**	81.93	89.59			benign	3165 6400

Table 7.4. The results of model with first convolution's stride 2

7.2.4 Model with Feature Map Doubled

This model is easy to understand, which is simply double the feature maps comparing with the normal model structure, the idea is inspired by [51], which claims that wider ResNet is helpful for image classification. Therefore, we want to know whether it works on histopathological images or not.

The model structure diagram is shown in (Figure 7.1, middle) and the detailed result can be found in Table 7.5, we can see that doubled feature maps can in deed increase the study capacity of model because almost all magnification's AUC increase, which is like 3×3 convolution model. Typically, sum becomes a pretty good aggregation method in this model.

magnification	**Image level**					**Batch level**			
	aggregation method	**accuracy (%)**	**precision (%)**	**recall (%)**	**F1 score (%)**	**accuracy (%)**	**confusion matrix**		
40×	sum	82.89	90.40	**84.88**	87.55	84.91	**AUC(%)**	84.15	
	vote	**86.28**	97.46	84.35	**90.43**				predict
	average	86.09	97.46	84.15	90.31				malignant benign
	exist	83.83	**99.44**	80.73	89.11		actual	malignant	16962 738
	exist3	84.59	99.15	81.62	89.54			benign	3275 5625
100×	sum	81.87	90.11	**83.46**	86.66	81.93	**AUC(%)**	82.91	
	vote	**83.30**	95.05	82.19	**88.15**				predict
	average	83.12	95.05	81.99	88.04				malignant benign
	exist	79.17	**99.18**	76.16	86.16		actual	malignant	17080 1156
	exist3	80.43	98.63	77.54	86.16			benign	3885 5779
200×	sum	**88.87**	99.18	**86.19**	92.03	88.14	**AUC(%)**	91.21	
	vote	88.69	**100.0**	85.48	**92.17**				predict
	average	88.50	**100.0**	85.28	92.06				malignant benign
	exist	85.58	**100.0**	82.21	90.23		actual	malignant	18218 32
	exist3	86.13	**100.0**	82.77	90.57			benign	3217 5933
400×	sum	**87.85**	95.64	**86.81**	**91.01**	86.56	**AUC(%)**	89.53	
	vote	86.73	98.55	83.70	90.52				predict
	average	86.54	98.55	83.50	90.40				malignant benign
	exist	84.67	**99.71**	80.90	89.32		actual	malignant	16833 396
	exist3	85.61	**99.71**	81.86	89.91			benign	3207 6364

Table 7.5. The results of model with feature maps doubled

7.2.5 Model with Two Pooling Layers Before Resnet

Our model faces a serious over-fitting problem, which will be introduced in detail in the following section 7.6.1. This model, with 2 pool layers before ResNet, is a try to solve the overfitting problem because we doubt that the study capacity of ResNet is so large that the net remembers all special features of train dataset, which results in over-fitting. Therefore, we want to apply more naïve convolutional layers, which has a smaller study capacity than ResNet, and less Residual blocks.

The model diagram is shown in Figure 7.1, right, and the results can be found in Table 7.6. We can see that almost all results have no difference from normal architecture, which means that Residual blocks are not the reason for overfitting.

magnification	**Image level**					**Batch level**			
	aggregation method	**accuracy (%)**	**precision (%)**	**recall (%)**	**F1 score (%)**	**accuracy (%)**	**confusion matrix**		
40×	sum	81.20	86.72	**85.28**	85.99	83.03	**AUC(%)**	79.76	
	vote	83.65	93.22	83.97	88.35				predict
	average	83.65	93.22	83.97	88.35				malignant benign
	exist	83.46	**99.44**	80.37	88.89		actual	malignant	16313 1387
	exist3	**84.21**	98.87	81.40	**89.29**			benign	3128 5772
100×	sum	80.79	89.29	82.70	85.87	83.40	**AUC(%)**	79.99	
	vote	84.38	96.15	**82.74**	88.95				predict
	average	84.02	95.60	82.66	88.66				malignant benign
	exist	**84.56**	**99.73**	79.08	88.21		actual	malignant	17386 843
	exist3	**84.56**	**99.73**	81.03	**89.41**			benign	3788 5883
200×	sum	**88.87**	98.36	**86.71**	92.17	88.15	**AUC(%)**	88.06	
	vote	88.69	**100.0**	85.48	92.17				predict
	average	**88.87**	**100.0**	85.68	**92.29**				malignant benign
	exist	84.67	**100.0**	81.29	89.68		actual	malignant	18181 69
	exist3	85.77	**100.0**	82.39	90.35			benign	3178 5972
400×	sum	**88.04**	97.38	**85.90**	**91.28**	86.10	**AUC(%)**	86.47	
	vote	86.17	99.13	82.77	90.21				predict
	average	86.36	99.42	82.81	90.36				malignant benign
	exist	83.36	**100.0**	79.45	88.55		actual	malignant	16971 256
	exist3	84.30	**100.0**	80.37	89.12			benign	3470 6103

Table 7.6. The results of model with 2 pool layers before ResNet

7.2.6 Normal Model with Dropout

Dropout is well known to be an effective way to solve over-fitting [52], therefore we also tried to apply dropout in our network. The model architecture is same as the normal model (Fig 7.3, left) , and we set up an additional dropout before the final fc layer with dropout rate 0.5. The result of normal model with dropout is in Table 7.7. The accuracy has a little improve comparing with the result of normal model, but we can still regard it as an effective method since almost all results are changing with a nice direction.

magnification	**Image level**					**Batch level**			
	aggregation method	**accuracy (%)**	**precision (%)**	**recall (%)**	**F1 score (%)**	**accuracy (%)**	**confusion matrix**		
40×	sum	81.58	87.85	**84.97**	86.39	82.94	**AUC(%)**	81.81	
	vote	**84.02**	93.79	84.05	**88.65**				predict
	average	**84.02**	93.79	84.05	**88.65**				malignant benign
	exist	81.77	**99.15**	78.88	87.86		actual	malignant	16469 1231
	exist3	82.71	98.02	80.32	88.30			benign	3306 5594
100×	sum	81.33	89.56	**83.16**	86.24	83.99	**AUC(%)**	82.38	
	vote	**84.92**	97.80	82.41	**89.45**				predict
	average	**84.92**	97.80	82.41	**89.45**				malignant benign
	exist	83.84	**99.45**	80.44	88.94		actual	malignant	17626 603
	exist3	84.20	**99.45**	80.80	89.16			benign	3863 5808
200×	sum	87.59	96.71	86.31	91.21	88.08	**AUC(%)**	87.02	
	vote	**88.87**	98.90	**86.36**	**92.21**				predict
	average	**88.87**	98.90	**86.36**	**92.21**				malignant benign
	exist	85.40	**100.0**	82.02	90.12		actual	malignant	17975 275
	exist3	86.86	99.73	83.68	91.00			benign	2992 6158
400×	sum	86.92	97.67	**84.42**	90.57	87.14	**AUC(%)**	84.77	
	vote	87.66	99.42	84.24	91.20				predict
	average	**87.85**	99.71	84.28	**91.34**				malignant benign
	exist	85.61	**100.0**	81.71	89.93		actual	malignant	17064 163
	exist3	86.36	**100.0**	82.49	90.41			benign	3284 6289

Table 7.7. The results of model with dropout

7.2.7 Overall Comparison among Different Model Structures

This part compares the results of different model architectures and the comparison is shown in Table 7.8.

magnification	Model architectures	image level			batch level	
		best aggregation method	accuracy (%)	F1 score (%)	accuracy (%)	AUC (%)
40×	normal	exist	81.20	89.59	82.93	82.19
	3×3 conv	vote	82.33	87.92	81.91	83.03
	stride 2	vote	**86.65**	**90.60**	**85.16**	**86.97**
	feature maps doubled	vote	86.28	90.43	84.91	84.15
	2 pools	exist3	84.21	89.29	83.03	79.76
	dropout	vote/average	84.02	88.65	82.94	81.81
100×	normal	exist3	84.74	89.39	83.37	76.98
	3×3 conv	exist	84.02	89.00	83.57	**84.91**
	stride 2	average	**86.00**	**90.10**	**84.66**	79.13
	feature maps doubled	vote	83.30	88.15	81.93	82.91
	2 pools	exist3	84.56	89.41	83.40	79.99
	dropout	vote/average	84.92	89.45	83.99	82.38
200×	normal	vote	88.87	**92.87**	**88.33**	85.02
	3×3 conv	average	**89.05**	92.34	88.31	89.78
	stride 2	average	87.77	91.46	87.19	85.07
	feature maps doubled	vote	88.69	92.17	88.14	**91.21**
	2 pools	average	88.87	92.29	88.15	88.06
	dropout	vote/average	88.87	92.21	88.08	87.02
400×	normal	exist3	86.73	90.62	86.15	82.61
	3×3 conv	average	87.29	90.93	86.82	**89.84**
	stride 2	average	86.36	90.15	85.68	87.09
	feature maps doubled	sum	87.85	91.01	86.56	89.53
	2 pools	sum	**88.04**	91.28	86.10	86.47
	dropout	average	87.85	**91.34**	**87.14**	84.77

Table 7.8. Overall results using slightly different model methods

From the table above, we can find that there are no huge differences comparing with the input pre-process because we adopted ResNet as our fundamental.

However, there are still some rules that can be found in the results, among

all models, we can conclude that dropout and feature maps doubled are helpful for classification no matter what the magnification is, and stride 2 has a huge improvement on dataset of magnification 40× and 100×. However, model with 2 pools, the contrast of feature maps doubled model, which reduces the complexity of model, do not get a performance boost. In comparison, we conclude that more complex structure can still make learn the features better.

7.3 Results of Different Segmentation Methods

Different segmentation methods will produce inputs of different size, which will absolutely be fed into different model architectures. Last section introduces the results of different model architectures, and the difference between these two parts is that the former one focused on the model architecture difference and kept input size same, while this section will mainly discussion the influence of different image segmentation methods.

When we study on the dataset, we found that tumor in low magnification images, such as 100×, was too small to be obviously found (Figure 7.3). We guess that the input size should be smaller when magnification is smaller to catch the local feature of tumor. To verify our guess, we implement and test our methods. Figure 7.5 shows the structure we used for different segmentation size and Table 7.9 is the overall comparison among different segmentation methods.

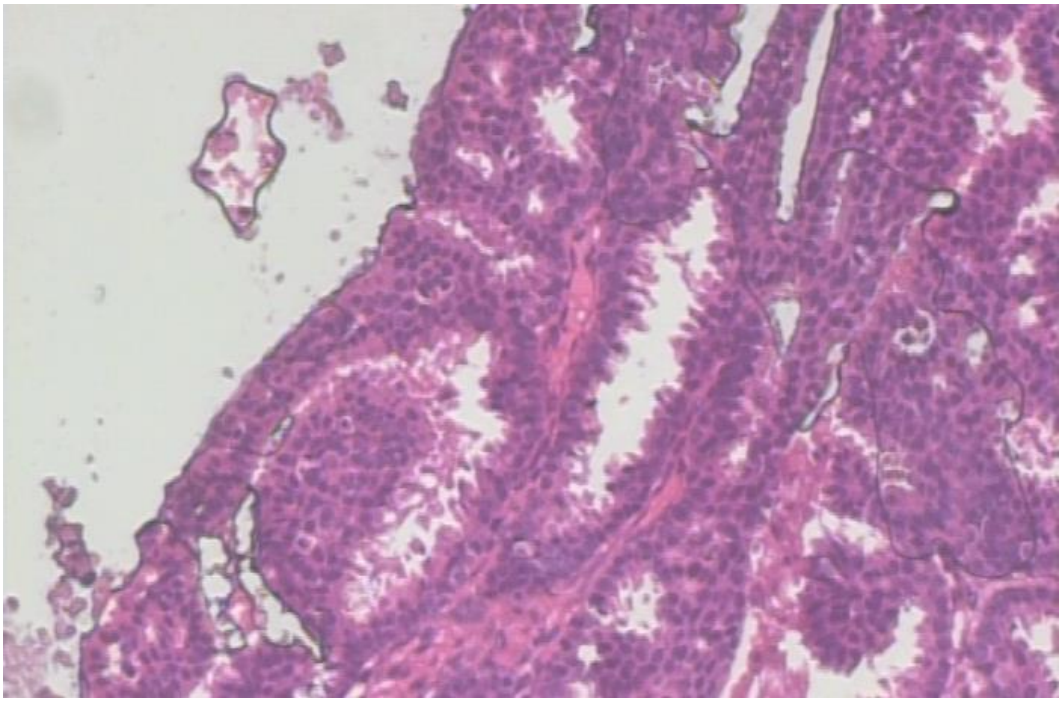

Figure 7.3 An example of 100× image, the tumor is too small

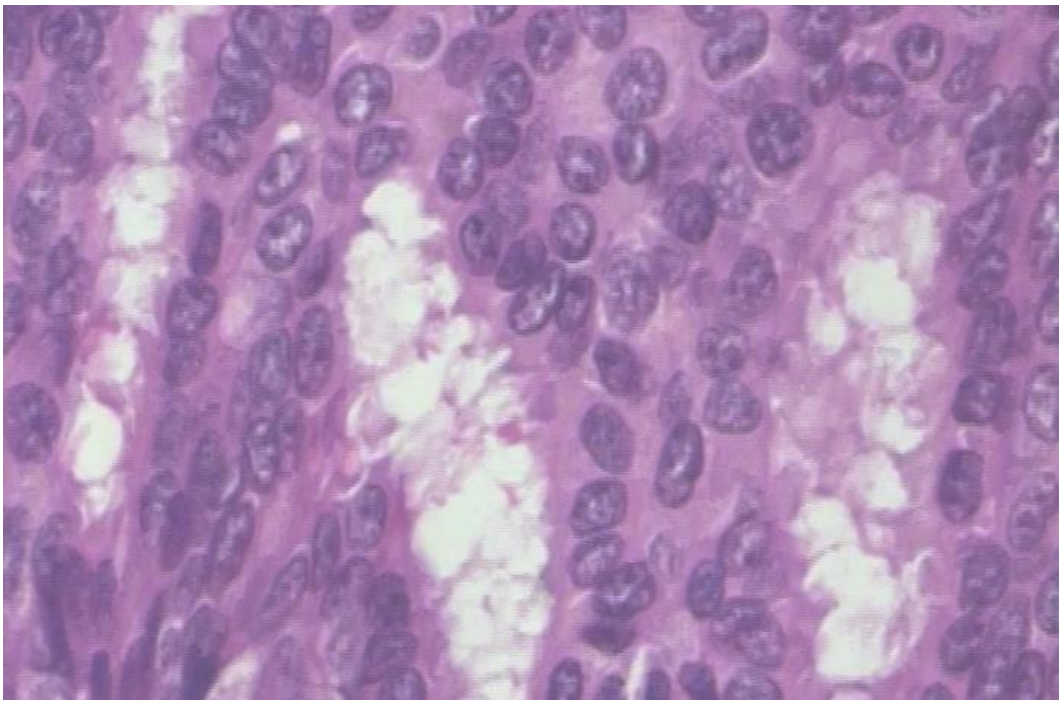

Figure 7.4 An example of 400× image, the tumor is obvious

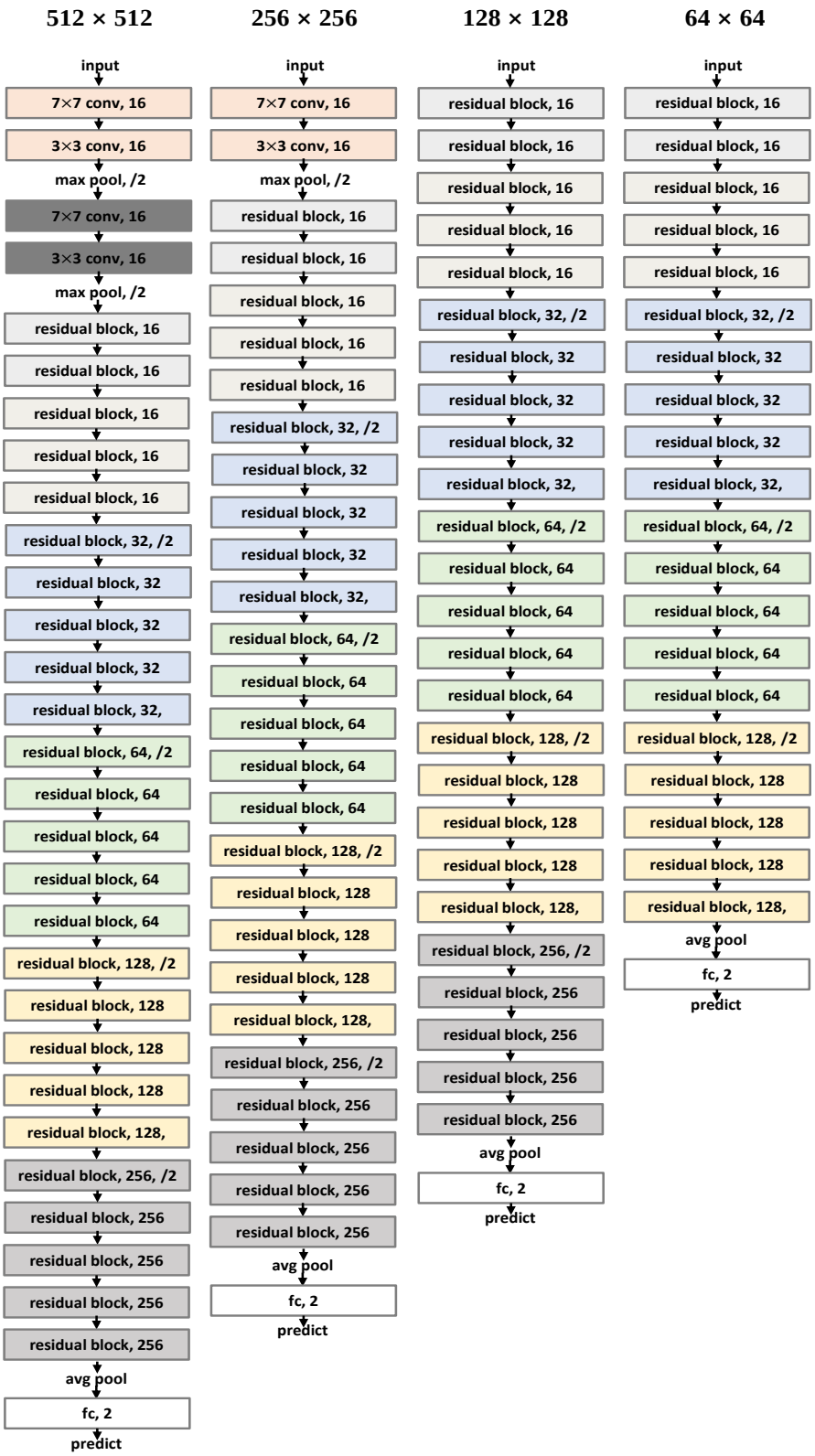

Figure 7.5 Structure for Different Segmentation Methods

magnification	Segmentation method	Input size	image level			batch level	
			best aggregation method	accuracy (%)	F1 score (%)	accuracy (%)	AUC (%)
40×	random	512×512	NA	NA	NA	NA	NA
	Random	256×256	exist	81.20	89.59	82.93	82.19
	random	64×64	vote	85.71	90.13	83.20	78.90
	sliding window	128×128	average	**87.41**	**91.15**	**84.56**	**82.69**
	sliding window	64×64	sum	85.34	89.54	83.88	82.12
100×	random	512×512	NA	NA	NA	NA	NA
	Random	256×256	exist3	84.74	89.39	83.37	76.98
	random	64×64	vote/average	**87.61**	**91.34**	**84.80**	81.61
	sliding window	128×128	vote/average	86.36	90.45	83.66	**86.65**
	sliding window	64×64	vote	86.89	90.86	84.40	83.86
200×	random	512×512	NA	NA	NA	NA	NA
	Random	256×256	vote	88.87	**92.87**	**88.33**	85.02
	random	64×64	sum	88.14	91.68	86.27	86.05
	sliding window	128×128	vote/average	**89.05**	92.41	87.10	86.42
	sliding window	64×64	average	88.50	92.06	86.84	**89.38**
400×	random	512×512	vote/average	**87.10**	90.71	**86.56**	85.26
	Random	256×256	exist3	86.73	90.62	86.15	82.61
	random	64×64	average	**87.10**	**90.76**	84.22	85.00
	sliding window	128×128	vote	86.91	90.72	84.37	85.78
	sliding window	64×64	vote/average	86.73	90.55	82.89	**86.81**

Table 7.9. Overall results using different image segmentation methods, segmentation method has been introduced in section 5.2 and 512×512 input size is too large to run correctly in 40×, 100× and 200× magnification factors.

According to Table 7.9, 64×64 and 128×128 ranks top 2 on both 40× and 100× test dataset while 256×256 and 512×512 dominates the results of 200× and 400× dataset, which is keeping with our guess. Also, we found that random segmentation method, which increases the variance of train dataset, is a little better than sliding window method.

7.4 Analysis

One of the advantages using DNN are that we needn't design a feature extractor by a medical expert, but instead the model will learn it by itself. Figure 7.6 displays the 16 feature maps learned on the first convolutional layer of our model. We can see that first convolution actually learned a edge detection rule by itself.

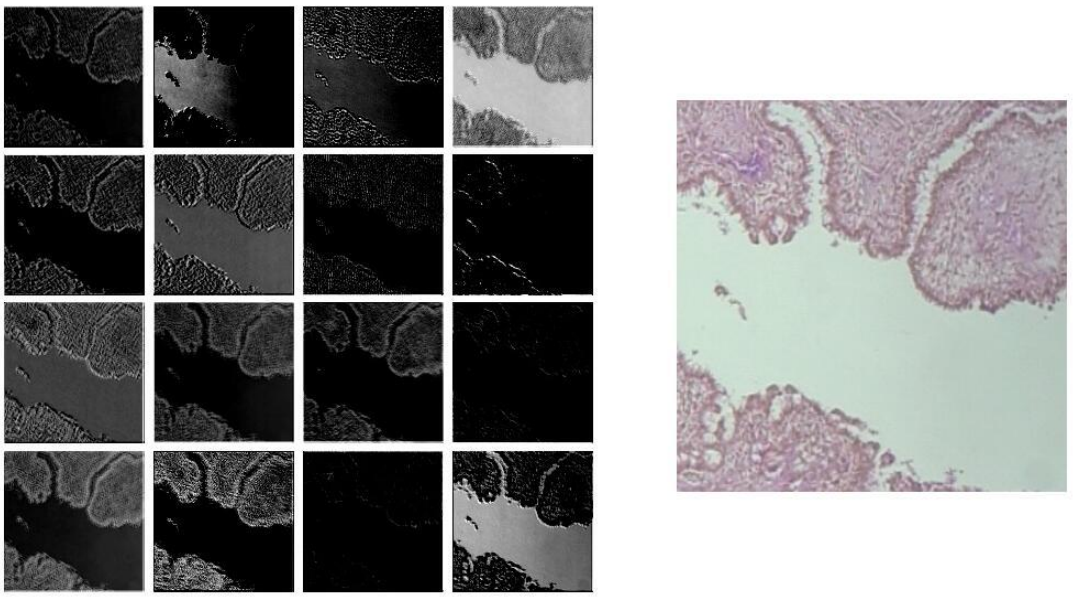

Figure 7.6 feature maps learned by first convolution layer, right side is the raw data and left side is 16 feature maps the model learned

From the idea of [57], we are able to visualize the location prediction of our model. We use the filter of last layer (shape 256×2) and the output of penultimate layer (shape 8×8×256) and implement a tensor-multiplication, after getting two feature maps with size 8×8, we resize the 8×8 image to input size, which is 256×256. Finally, we can use the resized image to visualize the local prediction to input of our model. Figure 7.7 shows an example of this kind of analysis.

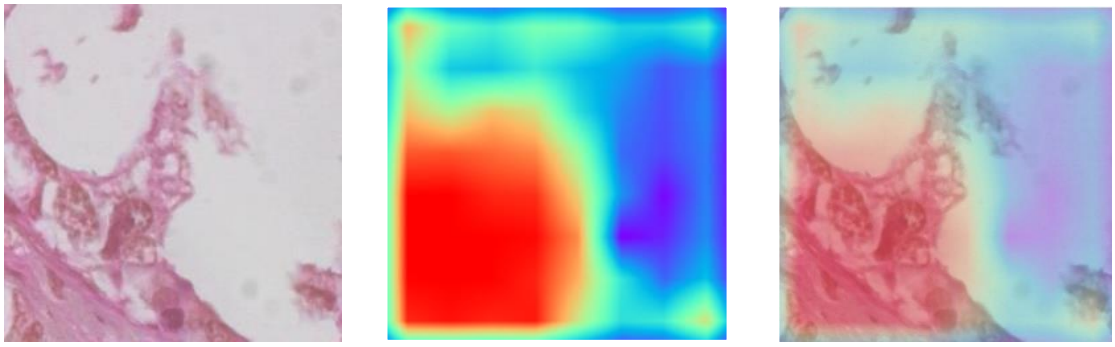

Figure 7.7 an example of localization prediction. Left: raw input. Middle: resized 256×256 prediction, red means more likely, blue means less likely. Right: the combination of two images before to visualize the result.

According to former experiments we have done, we can get a solid conclusion that datasets with different magnification factors need different hyper-parameters considering features of tumor. Typically, in this part, we implemented an "best" model combing former conclusions we got. We adapt the model architecture of Stride 2 (Fig 7.4, right), and add a dropout layer before the final fc layer. And CLAHE (section 5.2.6) is used to preprocess the data when magnification factor is not 40×, otherwise the preprocess method is CLAHE + whiten (section 5.2.5, section 5.2.6).

Table 7.10 shows the detailed results of "best" model and Fig 7.11 indicates one example of its ROC curve.

magnification	Image level					Batch level			
	aggregation method	accuracy (%)	precision (%)	recall (%)	F1 score (%)	accuracy (%)	confusion matrix		
40×	sum	**88.72**	97.74	**86.93**	**92.02**	86.80	**AUC(%)**	82.82	
	vote	87.41	99.44	84.41	91.30				predict
	average	87.78	99.44	84.82	91.55				malignant benign
	exist	81.58	**100.0**	78.32	87.84		actual	malignant	17522 178
	exist3	83.08	**100.0**	79.73	88.72			benign	3334 5566
100×	sum	84.92	94.78	**84.15**	89.15	85.22	**AUC(%)**	82.35	
	vote	**85.46**	97.90	82.38	**89.89**				predict
	average	**85.46**	97.90	82.38	**89.89**				malignant benign
	exist	82.94	**100.0**	79.30	88.46		actual	malignant	17959 273
	exist3	84.38	**100.0**	80.71	89.33			benign	3850 5818
200×	sum	88.50	98.08	**86.47**	91.91	88.50	**AUC(%)**	89.85	
	vote	**89.05**	99.73	86.05	**92.39**				predict
	average	88.87	99.45	86.02	92.25				malignant benign
	exist	86.31	**100.0**	82.95	90.68		actual	malignant	18043 207
	exist3	87.77	**100.0**	84.49	91.59			benign	2945 6205
400×	sum	**86.73**	93.60	**86.79**	90.07	90.43	**AUC(%)**	89.94	
	vote	86.17	96.51	84.26	89.97				predict
	average	86.35	96.80	84.30	**90.12**				malignant benign
	exist	85.61	**99.13**	82.17	89.86		actual	malignant	16494 739
	exist3	86.17	98.55	83.09	90.16			benign	2937 6630

Table 7.10. The results of "best" model whose parameters are selected manually to get good results

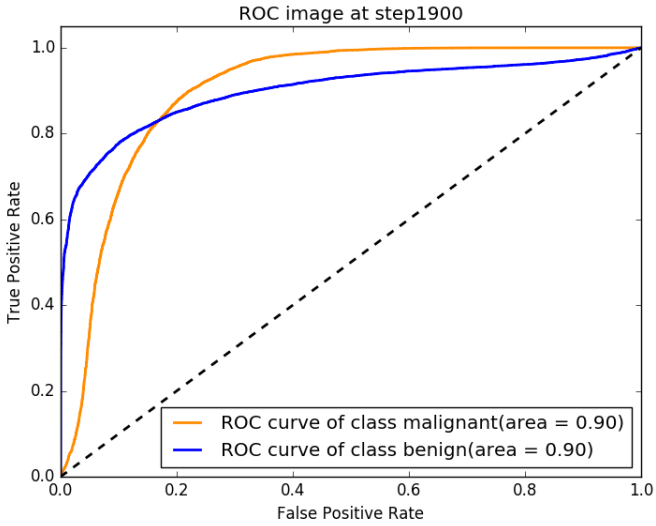

Figure 7.8 one example of ROC in our model results

We can obtain some general information about general result, aggregation methods and AUC value from these results above (comparison of different methods has been discussed above):

1. Our model achieves really high precision on image level, which is very practical because almost all malignant patients can be predicted as malignant.
2. Five aggregation methods we apply above have slightly different influence on results of image level, in summary, vote/average shows a better performance.
3. Lower magnification results have a lower AUC value, which means that more batches are labeled with not solid predictions. (Prediction of probabilities are closer to [0.5,0.5]). Therefore, we can conclude that lower magnification images have less information for learning.

magnification	Approach	Patient level	Image level	
		accuracy (%)	accuracy (%)	F1 score (%)
40×	[58]	83.00	NA	
	[40]	83.80	82.80	87.80
	[47]	**88.60**	**89.60**	**92.90**
	[56]	84.00	84.60	88.00
	This work	88.26	88.72	92.02
100×	[58]	83.10	NA	
	[40]	82.10	80.7	86.10
	[47]	84.50	85.00	88.90
	[56]	83.90	84.80	88.80
	This work	**88.17**	**85.46**	**89.89**
200×	[58]	84.60	NA	
	[40]	85.10	84.20	88.50
	[47]	85.30	84.00	88.70
	[56]	86.30	84.20	88.70
	This work	**92.27**	**89.05**	**92.39**
400×	[58]	82.10	NA	
	[40]	82.30	81.20	**86.30**
	[47]	81.70	80.80	85.90
	[56]	82.10	81.60	86.70
	This work	**90.34**	**86.73**	**90.12**

Table 7.11. Accuracy and F1 score compared with those presented in [58], [40], [47] and [56]

7.5 Comparison with Previous Works

Table 7.11 shows the overall comparison between our results and past paper's using same dataset.

Compared with accuracy and F1 score, which we defined earlier, our methods out-performs pervious work in [40], [47], [56] and [58] at both patient and image level generally. Our work is better than other research using same dataset in almost all of cases, only in the 40× zoom level our results are a little worse than previous best work. In the remaining cases, the accuracy and F1 score achieved at least 0.5% better, and the difference can be as large as 5% in most cases. Which means that our method is much better than pervious methods.

One guess for the reason of low accuracy at 40× zoom level, may be that images in low magnification factors, such as 40× and 100×, has a fewer information and features for model to catch and learn, this is what we conclude in last section. However, the advantage of our applied model, learn capacity, cannot make contribution to the result, which makes the results similar at 40× and 100× zoom level.

7.6 Limitation and Difficulties

Despite the result we get as aforementioned, we are also facing some limitation and difficulties. The following section will describe the problems and our proposed solutions.

7.6.1 Overfitting

We faced serious overfitting problem since we adopted ResNet. As we can see in Figure 7.9, the train accuracy can be easily up to 99% but the test accuracy is not as good as we expected.

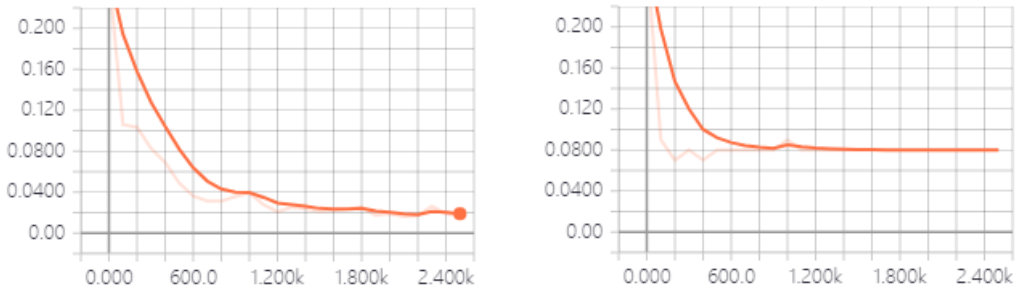

Figure 7.9 Train and validation accuracy comparison. Left: train error, which is close to 0 gradually. Right: validation error, which maintains at 0.08 level.

We have tried different technical to solve the problem, early stop, L2 regularization and dropout, all of them did not make a huge improvement but early stop can get an obvious increase, which can increase 2 to 3 percentage. We thought the reason may be the poor dataset, the dataset we used contain only 82 patients although there are thousands of images. We thought overfitting may also be the reason why past paper did not get a good-looking accuracy as well.

7.6.2 Out of Memory

Another difficulty we are facing now is the famous problem, OOM. ResNet consumed plenty of GPU memory due to the deep layers. Bigger input size will consume bigger memory and previous work of ResNet only fit an input with size 64×64 or 32×32.

But current input size our model adopts is 256×256, because malignant images can contain normal cells. If the image is divided into small patches such as 32×32, it is not guaranteed that tumor appears in all patches. Malignant patches without tumor become noise during training, and confused the network (Figure 7.10). For higher magnification and bigger crop size, this problem is less severe, as tumor cells will be larger and hence less likely to be missed.

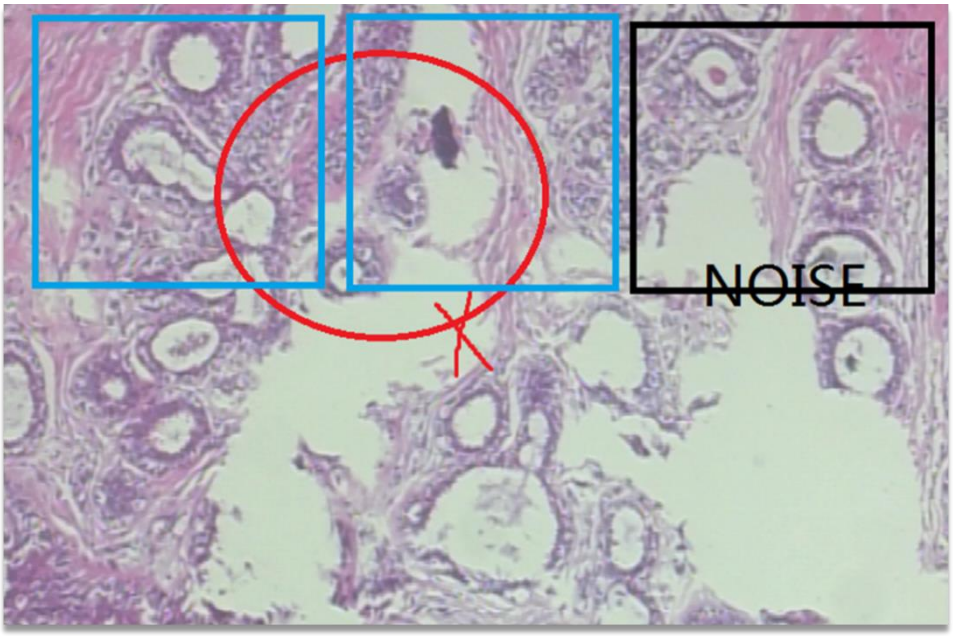

Figure 7.10. If red circle indicates a malignant tumor, then blue rectangle can be labeled as malignant correctly while black rectangle will become noise because there is no malignant tumor in it.

This is, therefore, the reason we build a traditional CNN above ResNet, we need a pool layer to implement down-sampling, which reduces the input size of ResNet to reduce memory allocation.

8 Conclusion

8.1 Term Review

When we started our final year project, we knew little about Tensorflow and histopathological image preprocess method or even some practical techniques in machine learning. We, therefore, regarded this project as a good chance for us to enhance our knowledge about deep learning and machine learning.

After continuous research and study from the related paper, we think we have achieved our basic goal, learning and understanding deep learning.

At the beginning we selected our project, we re-implemented the result of past papers with the help of professor Michael R. Lyu and his PhD student Zeng Jichuan, at the same time, we are looking for related paper on histopathological image preprocess and image classification by DNN.

With the successful re-implementation and our further understanding of DNN and histopathological image preprocess, we started to try designing our own model combing the feature of histopathological image and techniques using image classification using DNN. The final base model we use is ResNet, a state-of-the-art model in image classification with a top accuracy in general classification tasks.

But we did many adjustments to fit the histopathological images into ResNet model better, for example, adding a traditional CNN before ResNet to increase the size of model input, and we have tried different parameters or methods considering the feature of histopathological images and ResNet, all details and results can be found in Section Result. Finally, we achieved pretty high accuracy which was up to 90% average comparing with 86% average in past paper using same dataset.

8.2 Future Works

Our project is about breast cancer diagnosis using DNN and our primary goal is building a diagnosis system which help doctors to make decisions accurately and quickly. Therefore, our FYP is not only about histopathological image classification but also some other methods to help diagnosis.

During the study of Fast RCNN, which achieves satisfactory object detection accuracy [59], we find it possible to do object detection and image caption efficiently using current state-of-the-art methods. Therefore, our future work will be mainly about another model construction: building a high-accuracy DNN model using mammogram as input to do classification and tumor location detection due to the fact that our current task has an acceptable result comparing to the past paper.

Also, we will continue tuning our current models according to the shortcomings in Section Limitation we found.

9 Acknowledgements

Our deeper gratitude goes from first and foremost to our final year project advisor professor Michael R. Lyu and his PhD student Zeng Jichuan. Without their help and guidance, this research wouldn't be in the right path, especially professor Michael, who listened to our weekly presentation carefully every time and gave us many suggestions such as trying to do kinds of testing by controlling variables. Also, we are also extremely grateful for all authors in reference papers. This is our first time to truly understand the sentence by Newton: "If I have seen further, it is by standing on the shoulders of giants". We truly feel our lack of knowledge when standing on the shoulder of so many great engineering elites, which encourages us to be a great elite like them.

10 Reference

[1] J. G. Elmore, G. M. Longton, P. A. Carney, B. M. Geller, T. Onega, A. N. A. Tosteson, H. D. Nelson, M. S. Pepe, K. H. Allison, S. J. Schnitt, F. P. O'Malley, and D. L. Weaver, "Diagnostic Concordance Among Pathologists Interpreting Breast Biopsy Specimens," Jama, vol. 313, no. 11, p. 1122, 2015.

[2] [Online]. Available: https://en.wikipedia.org/wiki/AlphaGo_versus_Ke_Jie

[3] [Online]. Available: http://groups.csail.mit.edu/medg/index.html

[4] [Online]. Available: http://www.imperial.ac.uk/people/g.z.yang

[5] [Online]. Available: http://www.bioeng.nus.edu.sg/oeil/OCTAGON.html

[6] E. Alpaydin, Introduction to machine learning. Cambridge (Massachusetts): MIT Press, 2010.

[7] S. J. Russell and P. Norvig, Artificial intelligence: a modern approach. Upper Saddle River, NJ: Prentice Hall, 2003.

[8] B. E. Boser, I. M. Guyon, and V. N. Vapnik, "A training algorithm for optimal margin classifiers," Proceedings of the fifth annual workshop on Computational learning theory - COLT 92, 1992.

[9] Y. Lecun, Y. Bengio, and G. Hinton, "Deep learning," Nature, vol. 521, no. 7553, pp. 436–444, 2015.

[10] P. J. Werbos.Beyond Regression:New Tools for Prediction and Analysis in the Behavioral Sciences.PhD thesis, Harvard University, 1974.

[11] J. C. B. C. Schmidhuber, “Learning Complex, Extended Sequences Using the Principle of History Compression,” Neural Computation, vol. 4, no. 2, pp. 234–242, 1992.

[12] B. A. Olshausen and D. J. Field, “Emergence of simple-cell receptive field properties by learning a sparse code for natural images,” Nature, vol. 381, no. 6583, pp. 607–609, 1996.

[13] [Online]. Available: https://indico.cern.ch/event/510372/

[14] [Online]. "From not working to neural networking" The Economist. Available: https://www.economist.com/news/special-report/21700756-artificial-intelligence-boom-based-old-idea-modern-twist-not

[15] Shin HC, Orton MR, Collins DJ, Doran SJ, Leach MO. Stacked autoencoders for unsupervised feature learning and multiple organ detection in a pilot study using 4D patient data. IEEE Transactions on Pattern Analysis and Machine Intelligence. 2013;35:1930–1943.

[16] Roth HR, Lee CT, Shin HC, Seff A, Kim L, et al. Anatomy-specific classification of medical images using deep convolutional nets. Proceedings of IEEE International Symposium on Biomedical Imaging (ISBI) 2015

[17] L. G. Shapiro and G. C. Stockman, Computer vision. Upper Saddle River, NJ: Prentice Hall, 2001.

[18] Moeskops P, Viergever MA, Mendrik AM, de Vries LS, Benders MJNL, Iśgum I. Automatic segmentation of MR brain images with a convolutional neural network. IEEE Transactions on Medical Imaging. 2016;35:1252–1261.

[19] Zhang W, Li R, Deng H, Wang L, Lin W, et al. Deep convolutional neural networks for multi-modality isointense infant brain image segmentation. NeuroImage. 2015;108:214–224.

[20] Mao J, Xu W, Yang Y, Wang J, Huang Z, Yuille A. Explain images with multimodal recurrent neural networks. ArXiv.org Web site. [Accessed April 1, 2017]. https://arxiv.org/abs/1410.1090.

[21] Socher R, Karpathy A, Le QV, Manning CD, Ng AY. Grounded compositional semantics for finding and describing images with sentences. Trans Assoc Comput Linguist. 2014;2:207–218.

[22] Ciompi F, de Hoop B, van Riel SJ, Chung K, Scholten ET, et al. Automatic classification of pulmonary peri-fissural nodules in computed tomography using an ensemble of 2D views and a convolutional neural network out-of-the-box. Medical Image Analysis. 2015;26:195–202.

[23] Gao M, Bagci U, Lu L, Wu A, Buty M, et al. Holistic classification of CT attenuation patterns for interstitial lung diseases via deep convolutional neural networks. Computer Methods in Biomechanics and Biomedical Engineering: Imaging & Visualization. 2016;0:1–6.

[24] [Online]. Available: https://www.cbinsights.com/research/artificial-intelligence-startups-healthcare/

[25] M. Kowal, P. Filipczuk, A. Obuchowicz, J. Korbicz, R. Monczak, "Computer-aided diagnosis of breast cancer based on fine needle biopsy microscopic images", Computers in Biology and Medicine, vol. 43, no. 10, pp. 1563-1572, 2013.

[26] Y. M. George, H. H. Zayed, M. I. Roushdy, and B. M. Elbagoury, “Remote Computer-Aided Breast Cancer Detection and Diagnosis System Based on Cytological Images,” IEEE Systems Journal, vol. 8, no. 3, pp. 949–964, 2014.

[27] A. Esteva, B. Kuprel, R. A. Novoa, J. Ko, S. M. Swetter, H. M. Blau, and S. Thrun, “Dermatologist-level classification of skin cancer with deep neural networks,” Nature, vol. 542, no. 7639, pp. 115–118, 2017.

[28] Szegedy, C., Vanhoucke, V., Ioffe, S., Shlens, J. & Wojna, Z. Rethinking the inception architecture for computer vision. Preprint at https://arxiv.org/abs/1512.005672015

[29] V. Kumar, N. Fausto, A. K. Abbas, and R. N. Mitchell, Robbins basic pathology. Philadelphia, Pa: W.B. Saunders, 2008.

[30] J. P. Kösters and P. C. Gøtzsche, "Regular self-examination or clinical examination for early detection of breast cancer," Cochrane Database of Systematic Reviews, 2003.

[31] G. Majno and I. Joris, Cells, tissues, and disease: principles of general pathology. New York: Oxford, 2004.

[32] W. H. Wolberg, W. Street, and O. Mangasarian, "Machine learning techniques to diagnose breast cancer from image-processed nuclear features of fine needle aspirates," Cancer Letters, vol. 77, no. 2-3, pp. 163–171, 1994.

[33] [Online]. Available: https://cartesianfaith.com/2016/10/06/what-you-need-to-know-about-data-augmentation-for-machine-learning/

[34] [Online]. Available: https://opencv.org/about.html

[35] D. H. Hubel and T. N. Wiesel, "Receptive fields and functional architecture of monkey striate cortex," The Journal of Physiology, vol. 195, no. 1, pp. 215–243, Jan. 1968.

[36] van Tulder G, de Bruijne M. Combining generative and discriminative representation learning for lung CT analysis with convolutional restricted boltzmann machines. IEEE Transactions on Medical Imaging. 2016;35:1262–1272.

[37] Dou Q, Chen H, Yu L, Zhao L, Qin J, et al. Automatic detection of cerebral microbleeds from MR images via 3D convolutional neural networks. IEEE Transactions on Medical Imaging. 2016;35:1182–1195.

[38] K. He, X. Zhang, S. Ren, and J. Sun, “Deep Residual Learning for Image Recognition,” 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2016.

[39] Abadi, Martín, et al. "Tensorflow: Large-scale machine learning on heterogeneous distributed systems." arXiv preprint arXiv:1603.04467 (2016).

[40] F. A. Spanhol, L. S. Oliveira, C. Petitjean, and L. Heutte, “A Dataset for Breast Cancer Histopathological Image Classification,” IEEE Transactions on Biomedical Engineering, vol. 63, no. 7, pp. 1455–1462, 2016.

[41] [Online]. Available: https://datascience.stackexchange.com/questions/5224/how-to-prepare-augment-images-for-neural-network

[42] L. G. Hafemann, L. S. Oliveira, and P. Cavalin, “Forest Species Recognition Using Deep Convolutional Neural Networks,” 2014 22nd International Conference on Pattern Recognition, 2014.

[43] A. Kessy, A. Lewin, and K. Strimmer, “Optimal Whitening and Decorrelation,” The American Statistician, 2017.

[44] S. M. Pizer, E. P. Amburn, J. D. Austin, R. Cromartie, A. Geselowitz, T. Greer, B. T. H. Romeny, J. B. Zimmerman, and K. Zuiderveld, “Adaptive histogram equalization and its variations,” Computer Vision, Graphics, and Image Processing, vol. 39, no. 3, pp. 355–368, 1987.

[45] D. Attwell and S. B. Laughlin, “An Energy Budget for Signaling in the Grey Matter of the Brain,” Journal of Cerebral Blood Flow & Metabolism, vol. 21, no. 10, pp. 1133–1145, 2001.

[46] J. Kittler, M. Hatef, R. Duin, and J. Matas, “On combining classifiers,” IEEE Transactions on Pattern Analysis and Machine

Intelligence, vol. 20, no. 3, pp. 226–239, 1998.

[47] Spanhol, Fabio Alexandre, et al. "Breast cancer histopathological image classification using convolutional neural networks." Neural Networks (IJCNN), 2016 International Joint Conference on. IEEE, 2016.

[48] Deng, Lih-Yuan. "The cross-entropy method: a unified approach to combinatorial optimization, Monte-Carlo simulation, and machine learning." (2006): 147-148.

[49] Kohavi, Ron. "A study of cross-validation and bootstrap for accuracy estimation and model selection." Ijcai. Vol. 14. No. 2. 1995.

[50] L. Bottou, “Stochastic gradient tricks,” in Neural Networks, Tricks of the Trade, Reloaded, ser. Lecture Notes in Computer Science (LNCS 7700), G. Montavon, G. B. Orr, and K.-R. Müller, Eds. Springer, 2012, pp. 430–445.

[51] Zagoruyko, Sergey, and Nikos Komodakis. "Wide residual networks." arXiv preprint arXiv:1605.07146 (2016).

[52] Srivastava, Nitish, et al. "Dropout: a simple way to prevent neural networks from overfitting." Journal of machine learning research 15.1 (2014): 1929-1958.

[53] NVIDIA Corporation. (2015) Nvidia tesla product literature. [Online]. Available: http://www.nvidia.com/object/tesla_product_literature.html

[54] Powers, David Martin. "Evaluation: from precision, recall and F-measure to ROC, informedness, markedness and correlation." (2011).

[55] Hanley, James A., and Barbara J. McNeil. "A method of comparing the areas under receiver operating characteristic curves derived from the same cases." Radiology 148.3 (1983): 839-843.

[56] Spanhol, Fabio A., et al. "Deep Features for Breast Cancer Histopathological Image Classification."

[57] Oquab, Maxime, et al. "Is object localization for free?-weakly-supervised learning with convolutional neural networks." Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. 2015.

[58] N. Bayramoglu, J. Kannala, and J. Heikkila, “Deep learning for magnification independent breast cancer histopathology image classification,” in 23rd International Conference on Pattern Recognition, vol. 1, December 2016.

[59] Girshick, Ross. "Fast r-cnn." Proceedings of the IEEE international conference on computer vision. 2015.

11 Appendix

11.1 ResNet Function API

output_layer*(input_layer, num_labels,is_training,test)*

For creating the final layer which generates the prediction.

Args:

input_layer: 2D tensor

num_labels: Int. The number of output labels. (2 in our project)

Returns:

output layer, which is calculated by Y = WX + B.

batch_normalization_layer*(input_layer, dimension)*

For doing batch normalization

Args:

input_layer: 4D tensor to be normalized

dimension: Int. The depth of the 4D tensor, which is actually the number of feature maps in our project.

Returns:

the 4D tensor being normalized.

conv_bn_relu_layer*(input_layer, filter_shape, stride)*

For helping to do convolution, batch normalization and ReLU sequentially.

Args:

input_layer: 4D tensor

filter_shape: list of integers. The shape of filter.

stride: stride size for convolution

Returns:

4D tensor, which is calculated by Y = Relu(bn(conv(X)))

bn_relu_conv_layer*(input_layer, filter_shape, stride)*

For helping to do batch normalization, ReLU and convolution sequentially.

Args:

input_layer: 4D tensor

filter_shape: list of integers. The shape of filter.

stride: stride size for convolution

Returns:

4D tensor, which is calculated by Y = conv (Relu (conv(bn(X)))

residual_block*(input_layer, output_channel,*

is_training,first_block=False)

For defining one residual block (Image [])

Args:

input_layer: 4D tensor

output_channel: int. The number of output's feature maps.

first_block: Boolean value. If this is the first residual block in the whole network. (If yes, no down-sampling will be operated)

Returns:

4D tensor

inference*(input_tensor_batch, n, is_trainning = True)*

For defining the whole model structure of our project

Args:

input_*tensor*_batch: 4D tensor, which is actually [batch, image height, image width, image channels] in our project.

n: int. Number of residual blocks in each part with same number of output channels.

is_trainning: Boolean value. False if is testing, else True..

Returns:

last layer in the network, which is also the prediction of the model.

11.2 Tables of results using different preprocess methods

1. The results using RAW image as input (no preprocess method) in both batch level and image level
2. The results whose images are preprocessed by subtracting Gaussian image and applying CLAHE
3. The results using both CLAHE and whiten methods (keep the function order) in both batch and image level
4. The results using CLAHE in both batch level and image level
5. The results using both whiten and CLAHE methods (keep the function order) in both batch and image level
6. The results using whiten method in both batch level and image level
7. The results using preprocess method in past papers, simply demean the images
8. Overall results using different preprocess methods

magnification	Image level					Batch level			
	aggregation method	accuracy (%)	precision (%)	recall (%)	F1 score (%)	accuracy (%)	confusion matrix		
40×	sum	79.51	84.46	**84.70**	84.58	80.03	**AUC(%)**	80.68	
	vote	**81.95**	89.55	84.31	**86.85**				predict
	average	81.77	89.27	84.27	86.69				malignant benign
	exist	77.63	**93.79**	77.34	84.80		actual	malignant	15594 2106
	exist3	78.95	92.37	79.37	85.38			benign	3205 5695
100×	sum	77.56	82.42	**83.10**	82.76	79.09	**AUC(%)**	79.42	
	vote	77.56	87.91	79.80	83.66				predict
	average	77.56	87.91	79.80	83.66				malignant benign
	exist	78.28	**98.35**	75.69	85.54		actual	malignant	16097 2131
	exist3	**78.64**	96.98	76.57	**85.58**			benign	3981 5691
200×	sum	88.32	95.89	**87.72**	91.62	87.84	**AUC(%)**	88.36	
	vote	**88.87**	97.81	87.07	**92.13**				predict
	average	**88.87**	97.81	87.07	**92.13**				malignant benign
	exist	84.67	**99.45**	81.57	89.63		actual	malignant	17699 551
	exist3	86.13	99.18	83.22	90.50			benign	2782 6368
400×	sum	77.94	86.92	80.38	83.52	81.09	**AUC(%)**	85.73	
	vote	82.24	95.06	80.74	87.32				predict
	average	82.42	95.35	**80.79**	87.47				malignant benign
	exist	**82.99**	**99.13**	79.49	**88.22**		actual	malignant	16016 1219
	exist3	**82.99**	97.97	80.05	88.10			benign	3850 5715

Table 11.2.1 The results using RAW image as input (no preprocess method) in both batch level and image level

magnification	Image level					Batch level			
	aggregation method	accuracy (%)	precision (%)	recall (%)	F1 score (%)	accuracy (%)	confusion matrix		
40×	sum	62.40	65.54	**74.83**	69.87	65.09	**AUC(%)**	68.89	
	vote	65.22	84.75	69.61	76.43				predict
	average	65.79	85.59	69.82	76.90				malignant benign
	exist	**68.42**	**97.18**	68.52	**80.37**		actual	malignant	14944 2756
	exist3	67.48	95.20	68.37	79.57			benign	6529 2371
100×	sum	59.42	49.73	**80.80**	61.56	69.28	**AUC(%)**	70.39	
	vote	**69.12**	96.98	68.68	**80.41**				predict
	average	68.76	96.98	68.41	80.23				malignant benign
	exist	67.50	**98.90**	67.04	79.91		actual	malignant	17400 833
	exist3	67.68	98.63	67.23	79.95			benign	7739 1928
200×	sum	75.18	74.79	**86.12**	80.06	75.90	**AUC(%)**	81.52	
	vote	76.64	87.67	79.40	83.33				predict
	average	**77.19**	88.77	79.41	**83.83**				malignant benign
	exist	73.91	**96.99**	72.84	83.20		actual	malignant	15872 2378
	exist3	74.45	96.44	73.49	83.41			benign	4225 4925
400×	sum	77.76	81.40	**83.58**	82.47	78.26	**AUC(%)**	82.15	
	vote	**80.37**	90.99	80.88	**85.64**				predict
	average	80.18	91.28	80.51	85.56				malignant benign
	exist	71.78	**98.26**	69.98	81.74		actual	malignant	15433 1796
	exist3	74.21	**97.38**	72.20	82.92			benign	4031 5540

Table 11.2.2 The results whose images are preprocessed by subtracting Gaussian image and applying CLAHE

magnification	Image level					Batch level			
	aggregation method	accuracy (%)	precision (%)	recall (%)	F1 score (%)	accuracy (%)	confusion matrix		
40×	sum	85.90	95.48	**85.14**	90.01	86.17	**AUC(%)**	82.80	
	vote	**87.03**	99.15	84.17	**91.05**				predict
	average	86.84	99.15	83.97	90.93				malignant benign
	exist	82.89	**100.0**	79.55	88.61		actual	malignant	17374 326
	exist3	84.21	**100.0**	80.82	89.40			benign	3352 5548
100×	sum	78.64	86.54	**81.82**	84.11	81.44	**AUC(%)**	79.42	
	vote	81.87	93.68	81.38	87.10				predict
	average	**82.05**	93.96	81.43	87.24				malignant benign
	exist	81.15	**98.90**	78.09	87.27		actual	malignant	17085 1142
	exist3	81.69	98.08	78.98	**87.50**			benign	4035 5638
200×	sum	81.39	88.49	**84.33**	86.36	84.96	**AUC(%)**	85.41	
	vote	**85.77**	97.81	83.61	**90.15**				predict
	average	85.40	97.26	83.53	89.87				malignant benign
	exist	82.48	**99.73**	79.30	88.35		actual	malignant	17691 559
	exist3	83.58	99.45	80.49	88.97			benign	3562 5588
400×	sum	77.94	86.63	**80.54**	83.47	80.15	**AUC(%)**	82.05	
	vote	**80.75**	95.06	79.18	86.39				predict
	average	80.56	94.77	79.13	86.24				malignant benign
	exist	78.69	**99.13**	75.44	85.48		actual	malignant	16182 1047
	exist3	80.56	98.84	77.27	**86.73**			benign	4273 5295

Table 11.2.3 The results using both CLAHE and whiten methods (keep the function order) in both batch and image level

magnification	Image level					Batch level			
	aggregation method	accuracy (%)	precision (%)	recall (%)	F1 score (%)	accuracy (%)	confusion matrix		
40×	sum	81.58	86.72	**85.75**	86.23	82.93	**AUC(%)**	82.19	
	vote	82.89	93.50	82.96	87.92				predict
	average	**83.08**	93.50	83.17	88.03				malignant benign
	exist	81.20	**99.72**	78.10	**89.59**		actual	malignant	16542 1158
	exist3	82.89	99.15	79.95	88.52			benign	3382 5518
100×	sum	80.07	87.09	83.20	85.10	83.37	**AUC(%)**	76.98	
	vote	84.20	94.51	**83.50**	88.66				predict
	average	84.20	94.51	**83.50**	88.66				malignant benign
	exist	83.30	**98.90**	80.18	88.56		actual	malignant	17051 1185
	exist3	**84.74**	92.35	81.92	**89.39**			benign	3456 6208
200×	sum	87.77	96.99	**86.34**	91.35	88.33	**AUC(%)**	85.02	
	vote	**88.87**	99.73	85.85	**92.27**				predict
	average	88.69	99.73	85.65	92.15				malignant benign
	exist	86.31	**100.0**	82.95	90.68		actual	malignant	18128 122
	exist3	87.22	**100.0**	83.91	91.25			benign	3075 6075
400×	sum	83.18	91.86	83.60	87.53	86.15	**AUC(%)**	82.61	
	vote	**86.73**	97.67	**84.21**	90.44				predict
	average	**86.73**	97.67	**84.21**	90.44				malignant benign
	exist	86.17	99.71	82.45	90.26		actual	malignant	16743 487
	exist3	**86.73**	99.71	83.05	**90.62**			benign	3225 6345

Table 11.2.4 The results using CLAHE in both batch level and image level

magnification	Image level					Batch level			
	aggregation method	accuracy (%)	precision (%)	recall (%)	F1 score (%)	accuracy (%)	confusion matrix		
40×	sum	84.40	91.24	**86.13**	88.61	85.84	**AUC(%)**	80.97	
	vote	86.09	98.02	83.82	90.36				predict
	average	**86.28**	98.02	84.02	**90.48**				malignant benign
	exist	84.40	**100.0**	81.01	89.51		actual	malignant	17278 422
	exist3	84.02	99.44	80.92	89.23			benign	3345 5555
100×	sum	79.17	88.46	**81.31**	84.74	81.42	**AUC(%)**	82.23	
	vote	**82.23**	97.53	79.77	**87.76**				predict
	average	81.69	96.70	79.64	87.34				malignant benign
	exist	80.43	**100.0**	79.64	86.98		actual	malignant	17614 617
	exist3	81.51	99.73	78.06	87.58			benign	4566 5103
200×	sum	82.66	89.32	**85.34**	87.28	84.65	**AUC(%)**	87.63	
	vote	**85.22**	97.53	83.18	**89.79**				predict
	average	**85.22**	97.53	83.18	**89.79**				malignant benign
	exist	81.39	**99.73**	78.28	87.71		actual	malignant	17731 519
	exist3	82.85	99.18	79.91	88.51			benign	3686 5464
400×	sum	79.81	90.41	**80.57**	85.21	81.75	**AUC(%)**	83.38	
	vote	**82.80**	97.97	79.86	**87.99**				predict
	average	82.62	97.97	79.67	87.87				malignant benign
	exist	79.07	**99.42**	75.66	85.93		actual	malignant	16668 565
	exist3	80.19	**99.42**	76.68	86.58			benign	4325 5242

Table 11.2.5 The results using both whiten and CLAHE methods (keep the function order) in both batch and image level

magnification	Image level					Batch level			
	aggregation method	accuracy (%)	precision (%)	recall (%)	F1 score (%)	accuracy (%)	confusion matrix		
40×	sum	85.15	94.07	**85.17**	89.40	85.82	**AUC(%)**	78.65	
	vote	**86.84**	99.44	83.81	**90.96**				predict
	average	86.65	99.44	83.61	90.84				malignant benign
	exist	82.14	**100.0**	78.84	88.17		actual	malignant	17444 256
	exist3	83.46	**100.0**	80.09	88.94			benign	3516 5384
100×	sum	81.69	92.31	**81.95**	86.82	82.32	**AUC(%)**	82.19	
	vote	**83.12**	96.98	80.96	**88.25**				predict
	average	82.94	96.70	80.92	88.11				malignant benign
	exist	81.33	**99.73**	77.90	87.47		actual	malignant	17590 644
	exist3	82.05	98.90	78.95	87.80			benign	4288 5378
200×	sum	80.66	87.67	**83.99**	85.79	84.31	**AUC(%)**	86.22	
	vote	85.04	96.71	83.45	89.59				predict
	average	**85.22**	96.99	83.49	**89.73**				malignant benign
	exist	82.30	**99.73**	79.13	88.24		actual	malignant	17516 734
	exist3	83.58	99.45	80.49	88.97			benign	3564 5586
400×	sum	79.25	89.24	**80.58**	84.69	80.49	**AUC(%)**	83.11	
	vote	**81.31**	97.38	78.64	**87.01**				predict
	average	**81.31**	97.38	78.64	**87.01**				malignant benign
	exist	77.57	**99.13**	74.45	85.04		actual	malignant	16616 616
	exist3	78.69	98.84	75.56	85.64			benign	4613 4955

Table 11.2.6 The results using whiten method in both batch level and image level

magnification	Image level					Batch level			
	aggregation method	accuracy (%)	precision (%)	recall (%)	F1 score (%)	accuracy (%)	confusion matrix		
40×	sum	79.32	84.46	**84.46**	84.46	79.42	**AUC(%)**	82.41	
	vote	**79.51**	91.81	80.25	**85.64**				predict
	average	**79.51**	91.81	80.25	**85.64**				malignant benign
	exist	76.88	**96.33**	75.61	84.72		actual	malignant	16256 1444
	exist3	77.07	94.63	76.48	84.60			benign	4031 4869
100×	sum	72.35	75.27	81.07	78.06	78.01	**AUC(%)**	81.54	
	vote	78.10	87.64	80.56	83.95				predict
	average	78.54	87.64	**81.17**	84.28				malignant benign
	exist	78.99	**95.70**	77.02	85.75		actual	malignant	15927 2311
	exist3	**79.89**	95.33	78.51	**86.10**			benign	3823 5839
200×	sum	80.47	88.22	**83.42**	85.75	83.91	**AUC(%)**	82.62	
	vote	**84.67**	97.53	82.60	**89.45**				predict
	average	**84.67**	97.53	82.60	**89.45**				malignant benign
	exist	83.94	**98.90**	81.12	89.14		actual	malignant	17520 730
	exist3	84.12	98.63	81.45	89.22			benign	3680 5470
400×	sum	80.75	88.08	83.01	85.47	82.96	**AUC(%)**	82.97	
	vote	83.55	93.60	82.99	87.98				predict
	average	83.74	93.90	**83.03**	88.13				malignant benign
	exist	**84.67**	**98.84**	81.34	**89.24**		actual	malignant	16002 1237
	exist3	84.49	**97.38**	81.91	88.98			benign	3329 6232

Table 11.2.7 The results using preprocess method in past papers, simply demean the images