Computer-Aided Diagnosis of Low Grade Endometrial Stromal Sarcoma (LGESS)

Xinxin Yang
San Jose State University

Follow this and additional works at: https://scholarworks.sjsu.edu/etd_projects

Part of the Artificial Intelligence and Robotics Commons, and the Other Computer Sciences Commons

Recommended Citation
DOI: https://doi.org/10.31979/etd.44sj-cbwz
https://scholarworks.sjsu.edu/etd_projects/1007

This Master's Project is brought to you for free and open access by the Master's Theses and Graduate Research at SJSU ScholarWorks. It has been accepted for inclusion in Master's Projects by an authorized administrator of SJSU ScholarWorks. For more information, please contact scholarworks@sjsu.edu.
Computer-Aided Diagnosis of Low Grade Endometrial Stromal Sarcoma (LGESS)

A Project

Presented to

The Faculty of the Department of Computer Science

San José State University

In Partial Fulfillment

of the Requirements for the Degree

Master of Science

by

Xinxin Yang

May 2021
The Designated Project Committee Approves the Project Titled

Computer-Aided Diagnosis of Low Grade Endometrial Stromal Sarcoma (LGESS)

by

Xinxin Yang

APPROVED FOR THE DEPARTMENT OF COMPUTER SCIENCE

SAN JOSE STATE UNIVERSITY

May 2021

Dr. Mark Stamp Department of Computer Science
Dr. Thomas Austin Department of Computer Science
Dr. William Andreopoulos Department of Computer Science
ABSTRACT

Computer-Aided Diagnosis of Low Grade Endometrial Stromal Sarcoma (LGESS)

by Xinxin Yang

Low grade endometrial stromal sarcoma (LGESS) is rare form of cancer, accounting for about 0.2% of all uterine cancer cases. Approximately 75% of LGESS patients are initially misdiagnosed with leiomyoma, which is a type of benign tumor that is also known as fibroids. In this research, uterine tissue biopsy images of potential LGESS patients are preprocessed using segmentation and staining normalization algorithms. A wide variety of classic machine learning and leading deep learning models are then applied to classify tissue images as either benign or cancerous. For classic techniques, the highest classification accuracy we attain is 85%, while our best deep learning model achieves an accuracy of 87%. These results clearly indicate that properly trained learning algorithms can play a useful role in the diagnosis of LGESS.
I would like to thank Prof. Mark Stamp for serving as my thesis advisor. I would also like to thank Prof. Thomas Austin and Prof. William Andreopoulos for serving on my thesis committee. This work was made possible through data provided by the Cancer Imaging Archive. Lastly, I would like to thank all of my peers, classmates, and colleagues who befriended me and supported me throughout my masters education.
# TABLE OF CONTENTS

## CHAPTER

1. Introduction ................................................. 1

2. Background .................................................. 4
   2.1 Related work ........................................... 4
   2.2 Learning Algorithms ..................................... 6
      2.2.1 Multilayer Perceptron ............................. 6
      2.2.2 Random Forest ..................................... 6
      2.2.3 XGBoost ........................................... 7
      2.2.4 SVM ................................................. 7
      2.2.5 PCA .................................................. 8
      2.2.6 CNN .................................................. 8
      2.2.7 AlexNet ............................................. 8
      2.2.8 DenseNet ........................................... 10
      2.2.9 ResNet ............................................. 11

3. Dataset and Data Preprocessing ............................. 13

4. Experiments and Results .................................... 18
   4.1 Basic techniques ....................................... 18
      4.1.1 Learning with Multilayer Perceptron ................ 18
      4.1.2 Learning with Random Forest ....................... 19
      4.1.3 Learning with XGBoost ............................ 19
      4.1.4 Learning with SVM ................................ 20
4.1.5 Learning with PCA with SVM ....................... 20
4.2 Advanced techniques ..................................... 22
  4.2.1 Learning with basic CNN ......................... 23
  4.2.2 Learning with AlexNet ......................... 23
  4.2.3 Learning with DenseNet ....................... 24
  4.2.4 Learning with ResNet ......................... 25
  4.2.5 Learning with ResNet with realtime data augmentation .. 25
4.3 Discussion ................................................. 27

5 Conclusion ................................................. 29

LIST OF REFERENCES ........................................ 31
<table>
<thead>
<tr>
<th>Figure</th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Basic CNN structure</td>
<td>9</td>
</tr>
<tr>
<td>2</td>
<td>AlexNet architecture</td>
<td>9</td>
</tr>
<tr>
<td>3</td>
<td>DenseNet architecture</td>
<td>10</td>
</tr>
<tr>
<td>4</td>
<td>ResNet sub-network</td>
<td>11</td>
</tr>
<tr>
<td>5</td>
<td>Whole Slide Image Structure</td>
<td>13</td>
</tr>
<tr>
<td>6</td>
<td>Contour and Mask</td>
<td>15</td>
</tr>
<tr>
<td>7</td>
<td>Patches cut from one image</td>
<td>16</td>
</tr>
<tr>
<td>8</td>
<td>Color normalization for one image</td>
<td>16</td>
</tr>
<tr>
<td>9</td>
<td>Color normalization results 1</td>
<td>17</td>
</tr>
<tr>
<td>10</td>
<td>Color normalization results 2</td>
<td>17</td>
</tr>
<tr>
<td>11</td>
<td>Results of multilayer perceptron</td>
<td>19</td>
</tr>
<tr>
<td>12</td>
<td>Results of Random Forest</td>
<td>19</td>
</tr>
<tr>
<td>13</td>
<td>Results of XGBoost</td>
<td>20</td>
</tr>
<tr>
<td>14</td>
<td>Results of SVM</td>
<td>20</td>
</tr>
<tr>
<td>15</td>
<td>Results of PCA with SVM</td>
<td>21</td>
</tr>
<tr>
<td>16</td>
<td>Accuracy for each basic technique classifier</td>
<td>21</td>
</tr>
<tr>
<td>17</td>
<td>ROC comparison between above 5 classifiers</td>
<td>22</td>
</tr>
<tr>
<td>18</td>
<td>Basic CNN result</td>
<td>23</td>
</tr>
<tr>
<td>19</td>
<td>Test loss and accuracy with AlexNet</td>
<td>24</td>
</tr>
<tr>
<td>20</td>
<td>DenseNet performance</td>
<td>25</td>
</tr>
<tr>
<td>21</td>
<td>ResNet performance</td>
<td>26</td>
</tr>
<tr>
<td>Page</td>
<td>Section Description</td>
<td>Page</td>
</tr>
<tr>
<td>------</td>
<td>------------------------------------------------------------</td>
<td>------</td>
</tr>
<tr>
<td>22</td>
<td>ResNet with data augmentation performance</td>
<td>26</td>
</tr>
<tr>
<td>23</td>
<td>Accuracy for each advanced technique classifier</td>
<td>27</td>
</tr>
<tr>
<td>24</td>
<td>Performance of all models</td>
<td>28</td>
</tr>
</tbody>
</table>
CHAPTER 1

Introduction

Cancer is one of the most severe disease classifications threatening human life today [1]. It is the second leading cause of death in the United States, accounting for 21.6% of total deaths in a 2017 survey conducted by the Center for Disease Control (CDC) [2]. The tremendous medical costs of cancer treatments and the harm cancer brings to patients and their families makes cancer a necessary and important area of medical research.

Low grade endometrial stromal sarcoma (LGESS) is a tumor comprised of endometrial stromal cells. It is very rare, accounting for approximately 0.2% of uterine cancers [3, 4]. Most patients with LGESS have a good prognosis, with a 5-year survival rate of about 80% after surgical removal of the tumor. However, it has a relatively high recurrence rates of about 60%, and the disease-related death rate is estimated to be between 15% and 25% [5, 6].

When diagnosing LGESS, it is difficult to differentiate LGESS from benign leiomyoma, also known as fibroids. Only 10% of patients are correctly diagnosed with LGESS, whereas 75% are misdiagnosed with preoperative leiomyoma [7]. Many cases even remain misdiagnosed postoperative [8]. More accurate and automatic image analysis methods are needed to diagnose LGESS, assess treatment efficacy, and lower cancer-related costs to our healthcare system.

Designers of artificial intelligence (AI) algorithms for patient image analysis rarely possess the medical knowledge required for accurate modeling. The less than 100% accuracy of these algorithms make them more suited to aid in cancer risk assessment rather than definitive diagnostic tools [9]. Computers can reduce the workload of healthcare professionals by automating tedious tasks, such as tumor segmentation. Moreover, AI algorithms are more capable than humans at analyzing smaller, more
subtle structures in patient images [10]. Computers can analyze larger feature sets in a shorter amount of time, allowing for a more quantitative and nuanced analysis of images not easily perceivable to a human viewer. These capabilities have been showcased in a wide variety of use cases, including tumor segmentation, determination of tumor malignancy, and prediction of survivability in afflicted patients [11].

In this project, we apply machine learning and deep learning methods to classify soft tissue images of potential LGESS patients. We will apply these algorithms on a LGESS dataset procured from the Cancer Imaging Archive. This dataset has over 800 tissue biopsy images taken from 250 patients with uterine tumors (a tissue biopsy is never conducted unless the patient has a confirmed tumor). These tumors are classified as either cancerous or benign. All benign tumors located in the uterus are leiomyoma [12]. We will soon show how machine and deep learning algorithms can accurately differentiate LGESS tumors from their starkly similar fibroid counterparts with high accuracy, paving the way for a new state-of-the-art in LGESS diagnostic accuracy.

The remaining work is organized as follows: Chapter 2 is has a section to review literatures that discusses existing machine learning and deep learning algorithms that have been applied to cancer image analysis. The accuracy of these approaches, and the cancers they are applied to, are analyzed and compared. And it has another section to brief describe the machine learning and deep learning models we will experiment later in our project. Chapter 3 gives an overview of Whole Slide Imaging, the cancer image dataset used in this project, and the preprocessing strategies used on the data. This data preprocessing workflow includes Region Of Interest (ROI) segmentation, image patch extraction, and stain normalization. Chapter 4 details the machine learning and deep learning classifiers applied to our dataset. These classifiers include Multilayer Perceptron, Random Forest, XGBoost, SVM, PCA with SVM, CNN,
AlexNet, DenseNet, and ResNet. The performances of these models are compared and discussed. Chapter 5 concludes the paper, emphasizing again the importance of research on Computer-Aided Diagnosis for cancers like LGESS. It discusses the findings of our literature review and experiments, and forecasts possible future work to help diagnose LGESS and build effective treatment plans.
CHAPTER 2
Background

2.1 Related work

Machine learning has found widespread use in cancer classification and diagnosis [11, 13]. Although machine and deep learning research has been conducted on many different cancers, no studies exist pertaining to LGESS. This is most likely due to the rarity of LGESS compared to other cancers.

Mesrabad [14] applied Artificial Neural Networks (ANN), AlexNet, and Support Vector Machines (SVM) to classify prostate cancer. AlexNet is a deep learning method based on Convolutional Neural Networks (CNN). AlexNet yielded a classification accuracy of 86.3%, compared with 81.1% from SVM and 79.3% from ANN. Kharya et al. noted that ANN is the most widely used prediction technique in medical forecasting, but its structure is difficult to understand [15]. Their paper also lists the advantages and disadvantages of Decision tree, Naive Bayes, neural networks, and SVM methods for breast cancer detection.

Ashhar et al. researched the efficacy of deep learning algorithms for early detection of lung cancer. Previous research showed that lung cancer is always diagnosed at advanced stages, necessitating early and accurate screening [16]. They applied 5 state-of-the-art Convolutional Neural Networks to a dataset of Computer Tomography (CT) lung cancer images. These neural networks include DenseNet, GoogleNet, ShuffleNet, SqueezeNet and MobileNetV2. They analyzed the accuracy, sensitivity, specificity and ROC curves of these 5 models, and found that GoogleNet was the best CNN architecture for classifying CT images of lung cancer with an accuracy, specificity, sensitivity, and AUC of 94.53%, 99.06%, 65.67% and 86.84% respectively [16].

Vijayarajeswari et al. applied Hough transforms to mammogram images to detect features potentially symptomatic of breast cancer. These modified images were then
classified by SVM [17]. Their group attained a 94% accuracy with this strategy, far surpassing the classification accuracy of SVM on unmodified images. SVM’s classification accuracy was then tested on the Wisconsin Diagnostic Breast Cancer (WDBC) dataset, which is derived from biopsies of the breast [18]. Different sections of the dataset were analyzed with either linear, polynomial or RBF kernel functions. The accuracies of these three approaches were averaged, yielding a 99% classification accuracy on the WDBC dataset.

Ghoneim et al. conducted research on cervical cancer detection and classification, since cervical cancer is one of the leading causes of cancer death among women [19]. They extracted deep-learning relevant features from cervical cancer images using CNN’s model, then classified the images using extreme learning machines (ELM), multi-layer perceptrons (MLP) and autoencoder (AE)-based classifiers. The best performance came from their CNN-ELM-based system with a 99.5% accuracy on the 2-class detection problem and 91.2% accuracy on the 7-class classification problem [19].

Chaturvedi et al. proposed a classification method for skin cancer with better evaluation indicators than previous studies or dermatologists [20]. Their implementation of the MobileNet model performed with an overall accuracy of 83.1% for a 7-classes classification experiment. They believe their model could help dermatologists make decisions at critical stages.

Bharat applied traditional Machine learning classifiers such as K-Nearest Neighbor (k-NN), Naïve Bayes, Classification and Regression Trees (CART), and SVMs to predicting and diagnosing breast cancers [21]. Both [21, 22] conclude that these machine learning algorithms behave differently depending on the data set and parameter selection used for classification. In general, the k-NN technique has the best overall diagnostic effect, while Naïve Bayes and logistic regression have good performance when applied specifically to breast cancer diagnosis.
SVM is the best technique for recurrence/non-recurrence predictive analysis of breast cancer. Maglogiannis et al. proposed a SVM-based classifier for the prognosis and diagnosis of breast cancer. The optimized SVM algorithm performs well with high accuracy (96.91%), specificity (97.67%) and sensitivity (97.84%), outperforming the Bayesian and Artificial Neural Network classifiers it was compared against [23].

The results of these studies paint a promising picture for the predictive abilities of machine learning and deep learning in cancer medicine. AlexNet can be used to classify prostate cancers, while SVM has proven strength in predicting breast cancer malignancy [14, 23]. This project assess the strengths of deep learning and machine learning for diagnosing LGESS. We will include both AlexNet and SVMs in this study.

2.2 Learning Algorithms

Based on our literature reviews, we will include some basic machine learning models and some advanced deep learning techniques to experiment on our dataset. In this section, we will have a brief description of each model we will use in our project.

2.2.1 Multilayer Perceptron

Multi-layer Perceptron (MLP) is a kind of artificial neural network (ANN). It is generalized from the Perceptron Learning Algorithm (PLA). MLPs are also called Deep Neural Networks (DNN) due to their defining feature of multiple neuron layers. We call the first layer the input layer, the last layer the output layer, and the middle layer the hidden layer. The MLP does not specify the number of hidden layers, so you can choose the appropriate number of hidden layers for different needs. There is also no limit to the number of neurons in the output layer [24].

2.2.2 Random Forest

Random forest is an algorithm that integrates multiple decision trees through the idea of Ensemble Learning. Its basic unit is the decision tree [25]. Every decision tree
is a classifier, so for any given input sample, \( N \) trees will have \( N \) classification results. The random forest integrates all the category voting results and specifies the category with the most votes as the final output, which is the simplest idea of Bagging.

2.2.3 XGBoost

Extreme Gradient Boosting, otherwise known as XGBoost, was developed at the University of Washington by Dr. Tianqi Chen [26]. It was used in Kaggle’s Higgs subsignal recognition competition and has attracted wide attention because of its outstanding efficiency and high prediction accuracy.

XGBoost is essentially a Gradient Boosting Decision Tree (GBDT), but it tries to drive the speed and efficiency to the maximum possible. The core algorithm of XGBoost is to grow a tree by continually adding trees and doing feature splitting. Each time you add a tree, you learn a new function \( f(x) \) to fit the residual of the previous prediction. After we have completed our training and get \( k \) trees, XGBoost needs to add up the scores of the trees to get the final prediction score for a given sample.

2.2.4 SVM

Support vector machine (SVM) was first proposed by Vladimir N. Vapnik and Alexey Ya Chervonenkis in 1963, and the current version (Soft Margin) was proposed by Corinna Cortes and Vapnik in 1993 and was published in 1995 [27]. Before the emergence of deep learning in 2012, SVM was regarded as the most successful and best-performing machine learning algorithm of the decade.

SVM is a kind of binary classification model that maps the eigenvectors of an instance to some points in space. SVM "draws" a line that "best" distinguishes the two categories of points, so that if new points are created in the future, the line will still make a good classification. The basic model of SVM is defined as a linear classifier
with the largest interval in the feature space. The learning strategy of SVM is to maximize the interval. SVM also includes kernel tricks, which make it essentially a nonlinear classifier.

2.2.5 PCA

Principal Component Analysis (PCA) is one of the most widely used data dimensionality reduction algorithms [28]. Dimension reduction involves retaining some of the most important features of high-dimensional data while removing noise and unimportant features, thus improving data processing speed. The main idea of PCA is to map \( n \)-dimensional features to \( k \)-dimensional orthogonal features \((k < n)\), also known as main components.

2.2.6 CNN

CNN stands for Convolutional Neural Network. LeNet, created by Yann LeCun, is one of the earliest CNN structures used mainly for character classification problems. Convolutional Neural Network is a multi-layer supervised learning neural network. The convolutional layer and the pool sampling layer of the hidden layer are the core modules to realize the feature extraction function of Convolutional Neural Network. In this network model, the weight parameters in the network are adjusted layer by layer by using the gradient descent method to minimize the loss function, and the precision of the network is improved by frequent iteration training [29]. The following Figure 1 shows the structure of the basic CNN architecture.

2.2.7 AlexNet

AlexNet was developed by Alex Krizhevsky, a student of Dr. Geoffrey Everest Hinton (known as the father of neural networks). The AlexNet architecture is diagrammed in Figure 2 which is showed in [30].

It can be seen from the figure that the AlexNet structure has 8 layers in total.
The first 5 layers are convolutional layers, while the rest are fully connected layers. Paper [30] explains that there are 60 million learning parameters and 650,000 neurons in AlexNet. As explained in the paper, AlexNet runs on two GPUs. One GPU runs the upper layer-parts while the other runs the bottom layer-parts. After the first and second convolutional layers, there is a Region Proposal Network (RPN) layer. After the RPN layer and the fifth convolution layer there is the maximum pooling layer. Rectified Linear Units (ReLUs) come after each convolutional and fully connected layer.

From the paper we can see that AlexNet has the following features: ReLUs and dual GPU computing improve the training speed. These are applied to all convolutional layers and fully connected layers. Overlapping pooling layers are added to the first layer, second layer, and behind the fifth layer to improve accuracy and
make overfitting difficult. Local response normalization layers (LRN) are applied behind the first and second layers to improve accuracy. Lastly, dropout is applied to the first two fully connected layers to reduce overfitting.

2.2.8 DenseNet

The Dense Convolutional Network (DenseNet) was proposed by Gao Huang et al. in 2017 [31]. DenseNet connects each layer in a feed-forward manner. In normal convolutional neural networks, there are L connections to the L-layer network, while in DenseNet, there are L(L+1)/2 connections [31]. The input of each layer comes from the output of all previous layers. Figure 3 is the architecture diagram of DenseNet in [31].

![DenseNet Architecture](image)

Figure 3: DenseNet architecture

DenseBlock refers to the unique module of DenseNet. As shown in the figure above, all the front layers are densely connected to the back layers. In the same DenseBlock, the height and width of the feature layer will not change, but the number of channels will change. Transition Layer is a module that connects different DenseBlocks. The main function is to integrate the characteristics of the previous DenseBlock and reduce the width and height of the previous DenseBlock. In the Transition Layer, an AveragePooling2D(pooling) with a step of 2 is generally used to reduce the width and height of the feature layer [31].

DenseNet has the a few features such as it reduces the vanishing-gradient, it strengthen the feature delivery, it also encourages feature reuse, and it uses fewer parameters.
2.2.9 ResNet

ResNet was proposed and won the first place in the classification task of the ImageNet competition in 2015. ResNet is a residual network. We can think of it as a sub-network that can be stacked together to form a very deep network. It uses a connection method called "shortcut connection" [32].

Figure 4 is the sub-network structure of ResNet.

![Figure 4: ResNet sub-network](image)

We can see a "curved arc" in this diagram. This is the so-called "shortcut connection", referred to as identity mapping in the article. The whole structure is generally called a "building block", or Residual Block. Multiple similar Residual Blocks are connected in series to form ResNet [32]. As shown in Figure 4, a residual block has two paths: $F(x)$ and $x$. The path $F(x)$ conducts fitting on residuals and is called the residual path. The path $x$ performs identity mapping and is called the "shortcut". $F(x) + x$ composes the Residual Block.

The Residual Block could solve the "degeneration" problem of deep neural networks. We know that gradually superimposing layers on the network will improve the performance of the model, because the model is more complex, has stronger expressive ability, and can better fit the potential mapping relationship. While at the
same time, this architecture suffers from "degradation", meaning the performance drops rapidly if more layers are added to the network. The Residual Block could solve this problem by its "shortcut".
CHAPTER 3

Dataset and Data Preprocessing

This dataset was procured from The Cancer Imaging Archive, an organization offering datasets for cancer researchers all around the world [33]. It includes 888 uterine tissue biopsy images taken from 250 potential LGESS patients, formatted as SVS files. A separate annotations file contains the clinical diagnosis of each tissue image.

SVS is a file format for Whole Slide Images (WSI). In Whole Slide Imaging, a microscope slide is scanned to create a single high-resolution digital file [34]. Most WSIs have a resolution of 100,000 × 100,000 pixels. WSIs are usually stored in a pyramid structure. Each level of the pyramid holds different downsampled versions of the original image [35]. The more downsampled an image is, the less magnified it appears. Figure 5 shows us the pyramid structure of WSI images. In this project, we used the OpenSlide library, which provides methods to read and access WSI images stored in a variety of file formats, including SVS.

Figure 5: Whole Slide Image Structure

WSIs offer clear visualization of tumor characteristics, including tissue infiltration,
lymph node metastasis, and degree of differentiation. It is very helpful for the diagnosis, prognosis, grading, and staging of tumors [36].

The WSIs contained in the Cancer Imaging Archive dataset must undergo a color standardization stage before our classification algorithms can use them. Different production processes and scanning machines cause color variations in the WSIs. Images taken by different institutions, or by different operators at the same institution, will have different colors. These color differences can cause problems for algorithms that are not robust to these variations, even if said differences are imperceptible to the human eye.

The way to deal with this color difference is the standardization of colors. It is called color normalization or stain normalization. This involves normalizing all pictures to the color distribution of the same template picture [37].

Before applying color normalization to our WSIs, the Regions of Interest (ROIs) of each image need to be identified. This is done via standard Gaussian filtering and contour extraction techniques. These steps are explained in further detail below:

1. **Segmentation of the target region from the image.**

   (a) Use the OpenSlide library to read the level 2 image of the WSI. The second level of the WSI pyramid still has good resolution, but contains less data and is thus easier for OpenSlide to process. Save the level 2 image in the Portable Network Graphics (PNG) format.

   (b) Transform the image from the RGB color space to grayscale.

   (c) Apply a Gaussian filter to normalize the image. Make sure the filter’s threshold preserves the image’s contours.

   (d) Calculate the area of each contour and remove all contours that have an area below a certain threshold.
(e) Obtain the final image mask based on the contours acquired from the previous steps. This image mask comprises the tissue regions.

The Figure 6 below shows examples of extracted contours and resulting image masks. In each row, the first image is the original image in the dataset, the second image is the mask we got from the original image, and the last image is the original image with contours marked on it.

![Figure 6: Contour and Mask](image)

2. **Extract patches from an image’s target region.**

   Apply the previously generated mask to the image, then walk through the image and cut patches with a predefined length and width. All patches extracted must have a certain area occupied by the masked image. Patches below this area threshold consist mostly of empty space and are discarded.

   Examples of extracted patches are given below in Figure 7.

3. **Staining normalization.**
We applied Vahadane’s staining normalization method in our project to achieve color normalization [37]. The steps are as follows:

(a) Optical density calculation.
(b) Unsupervised staining density estimation.
(c) Color normalization.
(d) Normalized pixel intensity calculation.

Examples of images before and after color normalization are given below in Figure 8.

Figure 8 shows the color of an original image, labeled "Source". The image
labeled "Target" indicates the color we would like our image to become. The image labeled "Result" shows the original source image after color normalization. The color of this image is nearly identical to the target.

Figure 9 and Figure 10, given below, shows 12 images examples in our dataset before and after color normalization.

Figure 9: Color normalization results 1

Figure 10: Color normalization results 2

The images in the first rows of figures 9 and 10 are the raw source images. The images in the second rows of figures 9 and 10 are the color normalized versions of the images for the first row. Before normalization, the colors in our dataset can vary greatly. After our stain normalize process, all of these colors are normalized to the same color structure.
CHAPTER 4

Experiments and Results

After preprocessing the images and cutting them into patches, our final data set includes 4205 tumor images and 1459 normal images. The learning data is split in a stratified fashion using the train_test_split method from the sklearn package. We reserve 20% of the data as test data, while the remaining 80% data is used as training data. You could find the all codes for this project in my github [38].

The following metrics are used to evaluate the performance of our machine learning classifiers: Precision rate is the proportion of true positives in a sample to the amount predicted as positive. Recall measures how many truly positive samples can be predicted. In cancer detection, we hope to select models with high Recall rate. Finally, the F1 score is the harmonic average of precision and recall.

4.1 Basic techniques

In this section, we will discuss our experiment result on basic machine learning techniques such as Multiplayer Perceptron, Random Forest, XGBoost, SVM, and PCA with SVM.

4.1.1 Learning with Multilayer Perceptron

In this experiment, we imported MLPClassifier from the neural-network library of Scikit-Learn. We experimented about 10 different hyper-parameter combinations, and set the hyper-parameter which returns the best result. The fully connected neural network had 3 hidden layers. The first, second, and third hidden layers had 600, 800, and 300 neuron numbers, respectively. RELU is used as the activation function for each neuron. We also set it to train 600 epochs. The classification report for both the training data and the test data is listed in Figure 11.
4.1.2 Learning with Random Forest

We built our Random Forest model by importing the RandomForestClassifier algorithm from the sklearn.ensemble module. We tried 10, 50, 100, 150, 200, 300 as the number of trees in the forest, The accuracy didn’t grow much after 100. Therefore, we set 100 as the number of trees in the forest because it returns the best train and test accuracy. Figure 12 shows the classification performance of Random Forest.

**Figure 12: Results of Random Forest**

<table>
<thead>
<tr>
<th></th>
<th>Precision</th>
<th>Recall</th>
<th>F1_Score</th>
<th>Support</th>
</tr>
</thead>
<tbody>
<tr>
<td>Train</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1166</td>
</tr>
<tr>
<td></td>
<td>1</td>
<td>0.74</td>
<td>1.00</td>
<td>3363</td>
</tr>
<tr>
<td></td>
<td>Accuracy</td>
<td>0.74</td>
<td>0.85</td>
<td>4529</td>
</tr>
<tr>
<td>Test</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>292</td>
</tr>
<tr>
<td></td>
<td>1</td>
<td>0.74</td>
<td>1.00</td>
<td>841</td>
</tr>
<tr>
<td></td>
<td>Accuracy</td>
<td>0.74</td>
<td>0.85</td>
<td>1133</td>
</tr>
</tbody>
</table>

4.1.3 Learning with XGBoost

XGBoost was imported from the xgboost python package. We tested the XGBoost performance with 5 different hyper-parameter combinations, and chose the following as those returns the best accuracy. The parameter max_depth we set is 6, and the objective is binary:logistic. We obtained an accuracy of 85%. Figure 13 shows the classification performance of XGBoost.

**Figure 13: Results of XGBoost**

<table>
<thead>
<tr>
<th></th>
<th>Precision</th>
<th>Recall</th>
<th>F1_Score</th>
<th>Support</th>
</tr>
</thead>
<tbody>
<tr>
<td>Train</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1166</td>
</tr>
<tr>
<td></td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>3363</td>
</tr>
<tr>
<td></td>
<td>Accuracy</td>
<td>1</td>
<td>1</td>
<td>4529</td>
</tr>
<tr>
<td>Test</td>
<td>0</td>
<td>0.83</td>
<td>0.18</td>
<td>292</td>
</tr>
<tr>
<td></td>
<td>1</td>
<td>0.78</td>
<td>0.99</td>
<td>841</td>
</tr>
<tr>
<td></td>
<td>Accuracy</td>
<td>0.78</td>
<td>0.87</td>
<td>1133</td>
</tr>
</tbody>
</table>
4.1.4 Learning with SVM

We imported the SVM model from the Scikit-Learn library. We tested the SVC class with linear, rbf, polynomial, and sigmoid kernel functions. We observed nearly identical performance for all four kernels. The number of features in our image classification problem is large, so it is not surprising that a linear kernel would perform as well as more complex kernels. Since SVM with a linear kernel function is always faster to train and test, we decided to choose the linear kernel function for our SVM model. Figure 14 shows the classification performance of SVM.

<table>
<thead>
<tr>
<th></th>
<th>Precision</th>
<th>Recall</th>
<th>F1_Score</th>
<th>Support</th>
</tr>
</thead>
<tbody>
<tr>
<td>Train</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1166</td>
</tr>
<tr>
<td></td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>3363</td>
</tr>
<tr>
<td></td>
<td>Accuracy</td>
<td>1</td>
<td></td>
<td>4529</td>
</tr>
<tr>
<td>Test</td>
<td>0</td>
<td>0.83</td>
<td>0.51</td>
<td>0.64</td>
</tr>
<tr>
<td></td>
<td>1</td>
<td>0.85</td>
<td>0.96</td>
<td>0.90</td>
</tr>
<tr>
<td></td>
<td>Accuracy</td>
<td>0.85</td>
<td></td>
<td>1133</td>
</tr>
</tbody>
</table>

Figure 14: Results of SVM

4.1.5 Learning with PCA with SVM

In this experiment, we imported the PCA model from the sklearn.decomposition module. We set up a PCA model with 300 components to keep and a SVM model with the rbf kernel function. We then combined these two models with the make_pipeline method from the sklearn.pipeline module. Figure 15 shows the classification performance of PCA with SVM.

<table>
<thead>
<tr>
<th></th>
<th>Precision</th>
<th>Recall</th>
<th>F1_Score</th>
<th>Support</th>
</tr>
</thead>
<tbody>
<tr>
<td>Train</td>
<td>0</td>
<td>0.91</td>
<td>0.78</td>
<td>0.84</td>
</tr>
<tr>
<td></td>
<td>1</td>
<td>0.93</td>
<td>0.97</td>
<td>0.95</td>
</tr>
<tr>
<td></td>
<td>Accuracy</td>
<td>0.92</td>
<td></td>
<td>4529</td>
</tr>
<tr>
<td>Test</td>
<td>0</td>
<td>0.83</td>
<td>0.54</td>
<td>0.66</td>
</tr>
<tr>
<td></td>
<td>1</td>
<td>0.86</td>
<td>0.96</td>
<td>0.91</td>
</tr>
<tr>
<td></td>
<td>Accuracy</td>
<td>0.85</td>
<td></td>
<td>1133</td>
</tr>
</tbody>
</table>

Figure 15: Results of PCA with SVM
Figure 15: Results of PCA with SVM

Figure 16 compares the accuracy performance for each machine learning classifier. The accuracy of SVM and PCA with SVM is almost identical. The F1 score is almost the same as well. Thus, adding PCA to SVM for this data set does not make the model better. The XGBoost and SVM models have almost identical results on the test data, with SVM scoring only slightly higher.

<table>
<thead>
<tr>
<th></th>
<th>Precision</th>
<th>Recall</th>
<th>F1_Score</th>
<th>Support</th>
</tr>
</thead>
<tbody>
<tr>
<td>Train</td>
<td>0</td>
<td>0.90</td>
<td>0.77</td>
<td>1166</td>
</tr>
<tr>
<td></td>
<td>1</td>
<td>0.92</td>
<td>0.97</td>
<td>3363</td>
</tr>
<tr>
<td></td>
<td>Accuracy</td>
<td>0.92</td>
<td>0.92</td>
<td>4529</td>
</tr>
<tr>
<td>Test</td>
<td>0</td>
<td>0.82</td>
<td>0.54</td>
<td>292</td>
</tr>
<tr>
<td></td>
<td>1</td>
<td>0.86</td>
<td>0.96</td>
<td>841</td>
</tr>
<tr>
<td></td>
<td>Accuracy</td>
<td>0.85</td>
<td>0.85</td>
<td>1133</td>
</tr>
</tbody>
</table>

Figure 16: Accuracy for each basic technique classifier

The Receiver Operating Characteristic (ROC) curve and Area Under Curve (AUC) are often used to evaluate the quality of a binary classifier. The ROC curve is a curve plotted on the vertical axis of true positive rate (sensitivity) and abscissa of false positive rate (specificity). AUC is a probability value which is defined as the Area Under the ROC Curve. When a positive sample and a negative sample are randomly selected, the probability that the current classification algorithm will rank the positive
sample before the negative sample according to the calculated score is the AUC value. Therefore, the larger the value of AUC is, the more likely the current classification algorithm is to rank the positive sample before the negative sample value, which is conducive to better classification.

Figure 17 shows the ROC curves for the machine learning classifiers we experimented in our project. We can see that SVM has the max value of AUC. Based on the high accuracy and AUC, we conclude that SVM is the highest performing machine learning algorithm for our dataset.

4.2 Advanced techniques

In this section, we will discuss the performance of basic CNN, alexNet, denseNet, resNet, and resNet with real time data augmentation on our dataset.
4.2.1 Learning with basic CNN

We wanted to see the performance of the basic CNN with 2 convolution layers upon our dataset. We implemented our basic CNN model from tensorflow with 2 convolution layers, a learning rate of 0.005, a max_pool size of 2, and 1 final full connection. We experimented with 5 different batch sizes, generation numbers, and optimizers. The best results were acquired with the Adam Optimizer. These results are shown in Figure 18: Training accuracy: 81.0% Testing accuracy: 78.6%

![Figure 18: Basic CNN result](image)

4.2.2 Learning with AlexNet

Due to our computer’s hardware limitations, the Alex network we made uses a CPU for calculation. The principle of architecture of our AlexNet is the same as the original AlexNet proposed, but the calculation speed is slower. After tested with 5 different learning rate, loss function and optimizers, we used $1e^{-3}$ as the learning rate, softmax as the loss function, and Adam as the optimizer because this combination returns the best accuracy. We trained the model for 50 epochs. The best accuracy of
AlexNet we get for our dataset is 83%. The following Figure 19 shows test loss and accuracy.

![Figure 19: Test loss and accuracy with AlexNet](image)

4.2.3 Learning with DenseNet

We built a DenseNet based on the keras library. The convolution layer is initialized with a max pooling layer after it. We tested 3, 6, and 9 as the number of dense blocks in this network, they all returned the identical result. So, we set three dense blocks each with a transition layer follow after the convolutional layer. At the end we have another dense block and a classification layer. We had 3 experiments with different learning rate and filter combinations. We set the learning rate to 0.001 and the number of filters to 16 as a result of best accurate rate. We ran the model for 30 epochs. This DenseNet, when applied to our dataset, yields an accuracy of 85%. Figure 20 details DenseNet’s performance.
4.2.4 Learning with ResNet

We also used the keras library to build our ResNet. The ResNet is built with a convolution layer, max pooling layer, some basic block layers and an average pooling layer. The basic block layer includes a convolution layer, a batch normalization layer, and an activation layer. We ran our model for 20 epochs and obtained the best accuracy of 86%. Figure 21 shows the performance of ResNet on our dataset.

4.2.5 Learning with ResNet with realtime data augmentation

Since ResNet showed the best performance on our dataset, we decided to test ResNet’s capabilities in conjunction with realtime data augmentation. In realtime data augmentation, images in the existing dataset are copied, randomly rotated, shifted, or flipped, and augmented to the dataset. ResNet with data augmentation yielded an accuracy of 87%, as shown in Figure 22.

Figure 23 shows us the testing accuracy for each advanced technique we ex-
Figure 21: ResNet performance

Figure 22: ResNet with data augmentation performance

experimented with. From the figure we can see that the ResNet with real time data augmentation has the highest classification accuracy out of all other algorithms.
4.3 Discussion

Figure 24 summarizes the performance of the machine learning and deep learning algorithms on our LGESS image data. XGBoost, SVM and PCA with SVM all yield nearly identical 85% accuracy rates. All other machine learning algorithms used performed worse. While at the same time, SVM has the best AUC value which means it could rank more positive samples. Of all the deep learning methods we studied, ResNet yielded the best accuracy of 86%. ResNet’s accuracy is improved by an additional 1% when real time data augmentation is applied. Compared with these results, we could recommend to use ResNet with real time data augmentation for cancer classification of LGESS. But we could not recommend this model for every other type of cancer since the stark performance difference established algorithms can have on different diseases and datasets.
Figure 24: Performance of all models
CHAPTER 5

Conclusion

Cancer rates are increasing rapidly every year, incurring massive healthcare expenses and staggering death tolls [39]. Early detection of cancer can significantly reduce mortality and improve chances of survival. Giving healthcare providers fast, easy-to-use, high precision tools for automating cancer diagnosis will dramatically lower healthcare costs, give patients the treatment they need sooner, and ultimately save lives. Machine learning and deep learning algorithms are finding exciting applications in pathological image classification. These algorithms are beginning to play an important role in cancer detection and treatment planning, providing faster, more accurate tissue biopsy analyses than their human counterparts.

To date, the efficacy of machine learning and deep learning algorithms has not been tested on LGESS, most likely due to the rarity of this cancer. This project has uncovered exciting possibilities for these algorithms, showcasing their potential usability in the physician’s office. Our experiments found that XGBoost and SVM perform LGESS tissue image classification with 85% accuracy, while ResNet coupled with data augmentation demonstrates 87% accuracy. These results show the stark performance difference established algorithms can have on different diseases and datasets. The 85% accuracy of SVMs to our LGESS tissue images is meager when compared to its 99% accuracy in diagnosing breast cancer. These performance differences highlight the importance of choosing the correct learning algorithm for the cancer of interest.

Today, 75% of LGESS patients are incorrectly diagnosed with benign leiomyoma, leading to prolonged suffering, withholding of needed treatment, and lower chances of survival. The software demonstrated in this work can serve as a life-saving second opinion for physicians about to make an incorrect diagnosis. In future studies, more
accurate characteristics can be identified and extracted from these images, allowing our algorithms to work with a more refined and detailed feature space. Moreover, there exists a plethora of machine learning and deep learning algorithms that we did not study in this project. Expansions and continuations of this study stand to impact swathes of individuals suffering from this rare, unknown, and potentially deadly disease.
LIST OF REFERENCES


