ABSTRACT

A Deep Convolutional Neural Network Approach for Biomedical Applications

Hanh Hong Nguyen, Ph.D.

Mentor: Keith Evan Schubert, Ph.D.

Deep learning is a subset of machine learning that uses multi layer neural networks to perform desired tasks by using trained models. Neural networks are nonlinear mapping systems whose structure and function are loosely modeled on the physical structure of the nervous systems in humans and animals. In deep learning, convolutional neural networks (CNNs) have been used to analyze visual tasks for more than 40 years. Since the mid-2000s, they have revolutionized image processing and analysis. The goal of this dissertation is designing a deep CNN approach for biomedical applications, including automation of the process of colon polyps classification as well as single particle identification in radiation therapy.

A Deep Convolutional Neural Network Approach for Biomedical Applications

by

Hanh Hong Nguyen, B.S., M.S.

A Dissertation

Approved by the Department of Electrical and Computer Engineering

---

Kwang Y. Lee, Ph.D., Chairperson

Submitted to the Graduate Faculty of
Baylor University in Partial Fulfillment of the
Requirements for the Degree
of
Doctor of Philosophy

Approved by the Dissertation Committee

Keith Evan Schubert, Ph.D., Chairperson

Linda J. Olafsen, Ph.D.

Scott Koziol, Ph.D.

Jeffrey S. Olafsen, Ph.D.

Accepted by the Graduate School
December 2022

J. Larry Lyon, Ph.D., Dean

Page bearing signatures is kept on file in the Graduate School.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

ACKNOWLEDGMENTS

From the start, I am indebted to to my father and my mother. Thank you for the continuous love and support since day one I have received. I would like to thank the professors, staff and colleagues at the Department of Electrical and Computer Engineering depart who helped me through this journey, especially Mrs. Simcik and Dr. Linda Olafsen. Thank you my siblings, Ha and Hai, for keeping up with my craziness for the past years that we lived away from home but together at home. I would like to extend my sincere thanks to Holly and Paniz - the two big sisters I gained from coming to Texas. Lastly, I would like to express my deepest gratitude to Dr. Keith Evan Schubert for being the greatest force and guide I could ever ask for in my academic journey as well as my life.

DEDICATION

To ma, my Me.

CHAPTER ONE

Introduction

Machine learning is a subset of artificial intelligence. It uses statistics to enable computers to learn from experience by developing computational algorithms or models that can automatically infer hidden patterns from data. The goal of machine learning is to optimize a performance purpose by using example data or past experience [1]. The model may make a prediction in the future or gain knowledge from data, or both [1]. The core task of machine learning is to make inference from a sample by applying the theory of statistics in building mathematical models [1]. To achieve the desired purpose, in general, there is training as the first step and once a model is learned, its representation and algorithmic solution for inference needs to be efficient as well. In some cases the predictive accuracy and the efficiency of the learning algorithm are equally important [1]. There are three main types of machine learning, which are supervised learning, unsupervised learning, and reinforcement learning. In supervised learning, the model is trained with a labeled data set that contains the observations of an expert as well as the corresponding expected output labels. The goal of these models is to generate an inferred function that maps the feature vectors to the output labels. In unsupervised learning, the data set does not contain information about the output labels. The goal of these models is to derive the relationship between the observations and/or reveal the latent variables. In reinforcement learning, the learning a sequence of actions leads to a long-term reward.

Deep learning is a subset of machine learning that uses multi-layer neural networks to perform desired tasks by using trained models. Neural networks are nonlinear mapping systems whose structure and function are loosely based on the idea of the nervous systems in humans and animals.

A neural network consists of a large number of processors linked by weighted connections. The power of the system is from combining many processing units in a network. These units transform inputs linearly from other nodes and generate output as a single scalar by applying a nonlinear function [2]. The output is distributed to and acts as an input to the next layer in the network. The weights and biases used in the linear transformation are learned and updated through back propagation. Back propagation adjusts the network weights so the network produces the desired output in response to every input pattern in a predetermined set of training patterns [2]. The derivative chain rule is applied to calculate the derivatives of the network training error with respect to the weights. Thus the parameters are updated based on the gradients of the cost function in relation to the previous layer.

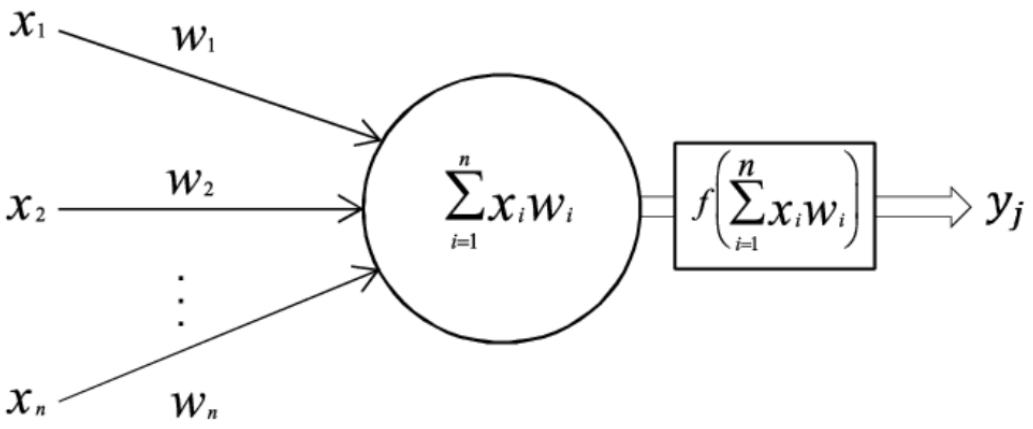

Figure 1.1: A diagram of a neuron

A neuron is a single unit of neural network which is illustrated in figure 1.1. This neuron is similar to neurons in the nervous system. The biological neurons consist of a cell body which connects to dendrites and axon. The axon transports the neural signal to other neuron at synapses which act as contact points [3]. In comparison to the biological neuron, the neural network neuron consists of inputs and outputs as the axons, the weight as synapses and the sum of the products of all input and associated weight as the dendrite.

Neural network neuron is a building block of a layer in a network. A single neuron has input x_i and has an associated weight w_i. A bias term is added to the summation $\sum x_i * w_i$ in the non-linear transfer function to transform the pre-activation level of the neuron to an output y_j [4]. The output y_j is the input to a neuron in the next layer. Several activation functions such as the rectifier function (rectified linear unit (ReLU)), the hyperbolic tangent function, the sigmoid function and the softmax function are available, which differ with respect to how they map a pre-activation level to an output value [4].

The primary purpose of this dissertation is to propose methods to apply deep CNNs to various biomedical application. To accomplish that, various CNNs models were investigated in three projects.

- Polyps classification
- Particle identification in radiation therapy
- Nitinol surgical wire

1.1 Contributions

In this dissertation, three ways are investigated to approach a biomedical application goal by applying a deep convolutional neural network. In each case, a different challenge was addressed and overcome. From a problem in which an simple, off the shelf pre-trained network can solve with a small data set in the first project of nitinol wire, to the second problem of lacking of intentional data in colonoscopy, and finally the problem of having many data but not the specific data for the right training in radiation therapy.

1.2 Outline

Chapter Two reviews the concept of convolution neural networks and the current state of the art of its application in the individual project. Chapter Three discusses about the polyp project. Chapter Four introduces a novel artificial intelligent approach for single particle identification in radiation therapy. Chapter Five presents the initial progress and

achievement for the nitinol surgical wire. Chapter Six discusses the conclusion and the future work.

CHAPTER TWO

Convolutional Neural Networks

2.1 Convolutional Neural Networks

Convolutional neural networks (CNNs) are one of the most popular types of deep neural networks applied to analyze visual data [5]. A CNN consists of many layers which includes a convolutional layer, a pooling layer, and a fully connected layer. Nearby pixels in an image are correlated with one another both in areas that exhibit local smoothness and areas consisting of structures such as edges of objects or textured regions. These correlations typically manifest themselves in different parts of the same image. Accordingly, instead of having a fully connected network where every pixel is processed by a different weight, every location can be processed using the same set of weights to extract various repeating patterns across the entire image. These sets of trainable weights, referred to as kernels or filters, are applied to the image using a dot product or convolution and then processed by a non linearity function. Each of these convolution layers can consist of many such filters resulting in the extraction of multiple sets of patterns at each layer. These convolution and pooling layers can be stacked to form a multi-layer network often ending in one or more fully connected layers as shown in Figure 2.1. The same concept can be applied in one-dimensional and three-dimensional (3D) to accommodate time series and volumetric data, respectively. CNNs contain fewer trainable parameters than in a fully connected neural network. They require less training time and fewer training data because of the reduction in the interaction of each neuron. It also reduces the likelihood of the model being over-fitted [6]. The architecture is specifically designed to take advantage of the presence of local structures in images, they are a natural choice for imaging applications. Over the years, tremendous progress has been made in image recognition with CNNs and

the availability of large data sets. However training CNN from scratch requires large data sets which are beyond what are available in most medical imaging situations [7].

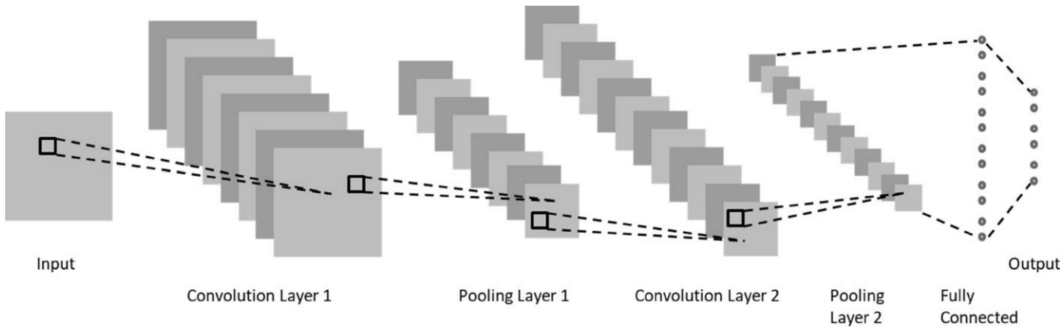

Figure 2.1: A diagram of a convolutional neural network

2.1.1 Convolution Function

Convolution layer is the basis of CNNs. A convolutional layer transforms the input by convolving it with a kernel and applying a nonlinearity to the output, known as a feature map. To understand the convolution layer, the convolution operation is the basis of that. Convolution operation can be explained as the operation that changes the shape of an original function into something else. Convolution operation is an orderly procedure where two sources of information are intertwined [8]. In mathematical terms, convolution operation involves two functions f and g that derives the third function to express how the shape of one is modified by the other. The convolution operation is expressed as following:

$$(f * g)(x) := \int_{-\infty}^{\infty} f(\tau)g(x-t)dt \tag{2.1}$$

Before CNN, convolution has been used in pure and applied mathematics applications such as numerical analysis, numerical linear algebra, and signal processing as well as image processing. For example, convolution is typically used to blur and sharpen images in image processing. Convolution is also used in design and implementation of finite impulse response filters in signal processing.

Convolutional layers apply a convolution operation to the input, passing the result to the next layer. The final output of the convolutional layer is a vector. Based on the type of problem we need to solve and on the kind of features we are looking to learn, we can use different kinds of convolutions.

A convolution operation converts all the pixels in its receptive field into a single value. When a convolution operation is applied to an image, the image size mathematically grows. For programming simplicity, the image size is defined to stay the same. Convolution brings all the information in the field together into a single pixel. A convolution operation is typically denoted with an asterisk.

Let's walk through one example of 2D convolution with a stride of 1 to clarify the definition of convolution, in which the result of the operation has the same size as the input image. Given the input matrix I,

$$I = \begin{bmatrix} 0 & 1 & 1 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 1 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 1 & 1 & 0 \\ 0 & 0 & 0 & 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 & 0 & 0 & 0 \\ 0 & 1 & 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 \end{bmatrix} \tag{2.2}$$

with a 3×3 kernel or feature detector k,

$$k = \begin{bmatrix} 1 & 0 & 1 \\ 0 & 1 & 0 \\ 1 & 0 & 1 \end{bmatrix} \tag{2.3}$$

The output of the convolution $I * k$ equals,

$$I * k = \begin{bmatrix} 0 & 2 & 2 & 3 & 1 & 1 & 0 \\ 1 & 1 & 4 & 3 & 4 & 1 & 1 \\ 0 & 1 & 2 & 4 & 3 & 3 & 0 \\ 0 & 1 & 2 & 3 & 4 & 1 & 1 \\ 1 & 1 & 3 & 3 & 1 & 1 & 0 \\ 1 & 3 & 3 & 1 & 1 & 0 & 0 \\ 0 & 2 & 1 & 1 & 0 & 0 & 0 \end{bmatrix} \tag{2.4}$$

The implementation of convolution in deep learning is desirable because of sparse interactions, parameter sharing and equivariant representations [8]. Sparse interactions, or as also known as sparse connectivity or sparse weights, happens when a kernel is smaller than the input. In this case, fewer parameters needed to be stored which can reduce the storage memory while improve the statistical efficiency by detecting smaller features. Parameter sharing occurs when certain parameters are used by more than one function in a CNN. Equivariant representations happens when the input changes, the output changes in the same way [8]. The convolution operation is equivariant to the function, it does not matter at which coordinates a feature occurs in the input matrix since it will still result in the same output after the convolution operation has been applied [8].

2.1.2 Pooling

Pooling performs a statistical summary over a window of outputs. A pooling layer aims to reduce the size of the feature map by taking the average or the maximum of small regions in the input. It often follows each convolution layer to both reduce the dimension and impose translation in-variance so that the network becomes immune to small shifts in location of patterns in the input image. A max pooling with 2×2 window size where the output is the maximum value within a window is often used in CNN.

2.1.3 Fully Connected Layer

A fully connected layer is placed before the output (last) layer to tune the weights and biases to create a stochastic likelihood representation of each class based on the activation maps created by the previous layers.

2.2 Transfer Learning

To train a deep CNN from scratch typically requires huge data set, well beyond what is available in most medical imaging settings. Therefore, transfer learning is one of the most practical schemes to solve that problem. Transfer learning helps to address the issue as the knowledge from pre-trained models to solve one task can be transferred to accomplish another task.

Transfer learning helps to address the issue as the knowledge from pre-trained models to solve one task can be transferred to accomplish another task. There are two types of transfer learning: (i) extracting features from a well trained CNNs model over a large data set of images then adding a new separate classifier on top of the learned feature maps and (ii) fine tuning the last layers of a pre-trained CNNs model by training newly added specialized classifier layers. In transfer learning, the neural network is trained on regular images, from large databases of images. The initial layers of the resulting neural networks can recognize general features like lines, corners, etc. In general, as you move through the layers you move from basic features to more complex features by aggregating the results of the earlier layers. The last layers of these networks are thus specialized to the image sets they were trained on. We remove the last several layers and then train on our image data set for the intended purpose. Since the low and middle level feature recognition is already well tuned, the new final layers can be learned from a few thousand images.

In a supervised learning scenario, the labeled data set, or the ground truth data set, is utilized for training a neural network. After training the neural network, the prediction model can be applied to mixed radiation field data to identify protons from other particles, for example.

When a model that has been trained for a task is reused to as the beginning model for a different task. A pre-trained network is loaded to use as a starting point of a new task. The early layers are used to distinguish low level features: edges, colors, etc.. to replace the last layers. The final layers of the network are tuned to distinguish features specific to your data set. The network is then trained by importing the new images into the network The network is tested by using a different set of images, test the accuracy. Results are utilized by fine tuning the parameters of the final layer

2.2.1 VGG-16

VGG-16, developed by Simonyan and Zisserman of the Visual Geometry Group from the University of Oxford, is one of the pre-trained CNNs models [9]. VGG achieved good results in both image classification and localization at the 2014-ILSVRC (Imagenet) competition. VGG-16 is made up of thirteen convolutional layers and three fully connected layers. All convolutional and fully connected layers have a ReLU activation function applied to their output except for the last layer which has a softmax activation function applied to it. The smallest size of 3×3 filters are included in the model to help with learning more complex features by increasing the depth of the network [2]. The main limitation of VGG-16 is the number of 138 million parameters which can become a burden to computation resources.

2.3 Generative Adversarial Network

A generative adversarial network (GAN) consists of a generator that produces new data for training and a discriminator that determines the probability of whether the generated candidates belong to the training samples or not [10].

Figure 2.2 shows an overview of a general GAN. The generator network produces new data instances that try to mimic the data used in training, while the discriminator network tries to determine the probability of whether the generated candidates belong to the training samples or not. The two networks are trained jointly with back-propagation,

with the generative network becoming better at generating more realistic samples and the discriminator becoming better at detecting artificially generated samples. GANs have recently demonstrated great potential in medical imaging applications such as image reconstruction for compressed sensing in magnetic resonance imaging [10].

Model collapse, vanishing gradients, problem with counting and problem with perspective are the major challenges of GAN. Model collapse happens when the generator continually produces a similar image and discriminator unable to consider the difference in the generated image. It prevents the generator from learning the whole image and just focuses on a particular area. Vanishing gradients occurs when the gradient loss function approaches zero. The discriminator does not give enough feedback to the generator, hence, the generator cannot produce reliable sample. Problem with counting used to occur in early GANs, in which the generator cannot produce the right amount of details on the sample image. Problem with perspective happens when the generator cannot differentiate between the front and back view in the original image.

2.4 Validation

Cross validation is a statistical method to evaluate the predictive validity of linear regression equations used to forecast a performance criterion from scores on a battery of tests [11]. It compares learning algorithms by dividing data into two segments: one used to learn or train a model and the other used to validate the model. Testing was done using a 10-fold cross validation. The image data was randomly split into 10 groups ensuring that all images of a polyp stayed in the same group. Ten different neural networks were trained by removing 1 of the 10 groups and training on the remaining 9 groups. The group that was removed was then used for testing. This methodology allows a large training set for the neural network, while still allowing all the data to be used for testing results by aggregating the different systems.

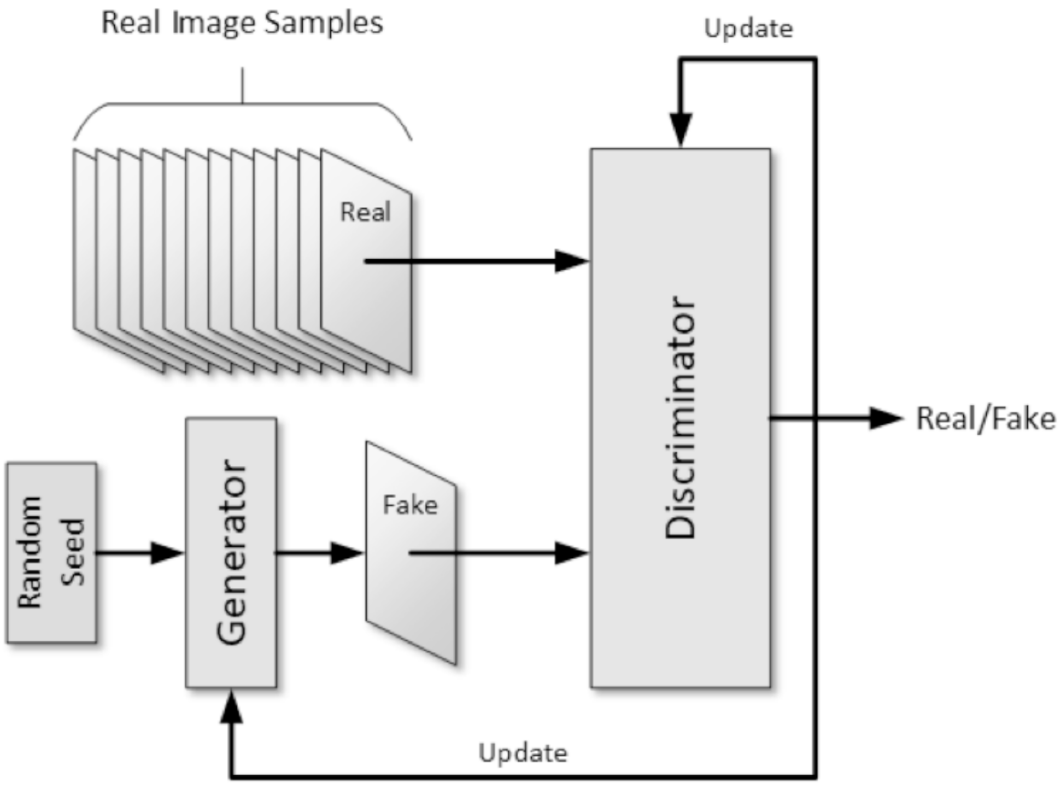

Figure 2.2: A diagram of a generative adversarial network

2.5 Confusion Matrix

Accuracy is the first and most common metric to evaluate the performance of a classification task. Accuracy is the proportion of correct predictions among the total number of predictions over the total population. However accuracy cannot show the detail performance of one task because if only based on accuracy only, one model can consistently or completely identify incorrectly one class and it is unnoticed because the overall performance is good. Hence, there is a need for a metric that shows a relationship of other metrics in an identification task. A confusion matrix is a table that is used to define and analyze the performance of a classification task. The basic confusion matrix consists of four basics numbers with a configuration as in Figure 2.3:

(1) True positive (TP) is an outcome where the model correctly predicts the positive class. It means that the identified cases in this class tested positive and truly positive.

(2) True negative (TN) is an outcome where the model correctly predicts the negative class. It means that the identified cases in this class tested negative and truly negative.

(3) False positive (FP) happens when a positive result for a test is detected when it should have been a negative result. It is a type I error.

(4) False negative (FN) happens when a negative result for a test is detected when it should have been a positive result. It is a type II error.

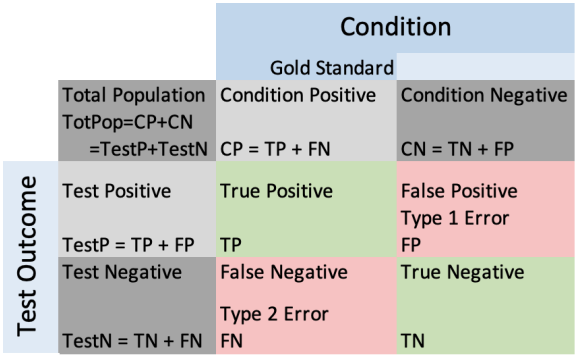

Figure 2.3: A diagram of a general confusion matrix

As noted above, all the diagonal components denote the correct classified items. The better classifier is the closer to zero the off diagonal components of confusion matrix it has. The effectiveness of a classifier is measured through the values that generated in the confusion matrix beside accuracy. An expanded confusion matrix also generates a visualization of metrics like accuracy, sensitivity and specificity.

The following Figure 2.4 is the example of the full confusion matrix used in this study. Beside the core TF, TN, FP, and FN, this matrix includes the accuracy, sensitivity, negative predictive value, positive predictive value, miss rate, fall-out, positive likelihood

ratio, negative likelihood ratio, false omisson rate, false discovery rate, and diagnostic odds ratio.

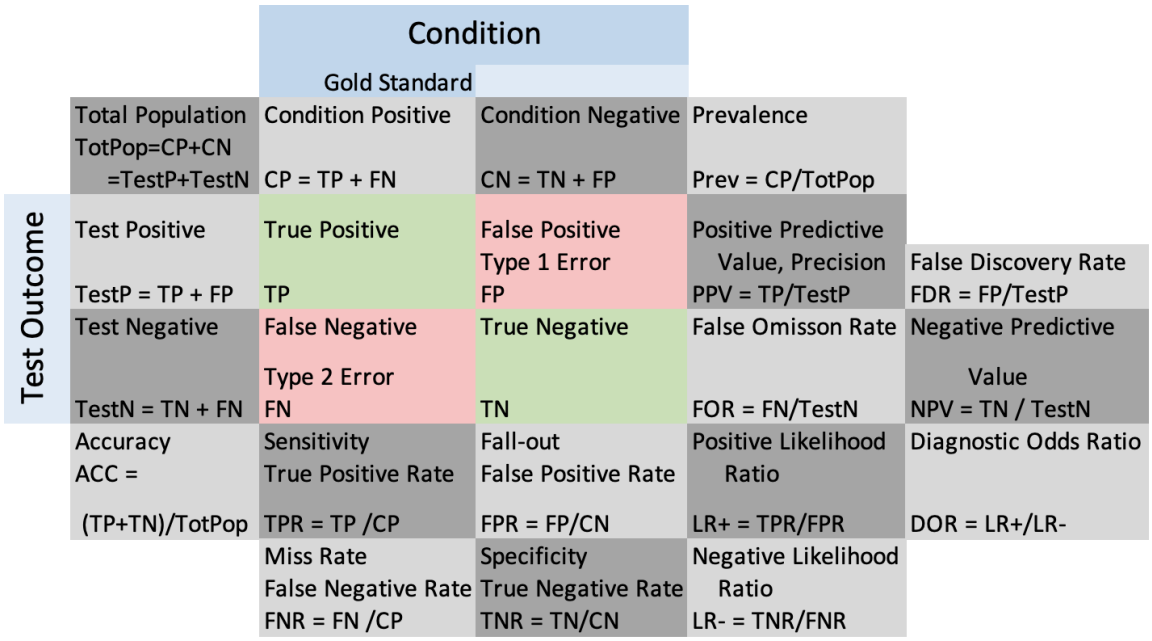

Figure 2.4: A diagram of a full confusion matrix

Sensitivity is the proportion of positively classified cases among the total number of positive cases. It is the rate of genuine positive predictions or the the ability to designate an individual with disease as positive. A highly sensitive model is mostly to rule out the negative tests, which means that there are few false negative results, and thus fewer cases of disease are missed. Therefore, a sensitive model is when negative rules out the disease, or SNOUT.

Specificity is the proportion of negatively classified cases among the total number of negative cases. The specificity of a test is the ability to designate an individual who does not have a disease as negative. A highly specific model can rule out the negative tests with high confidence. Specific model is when positive rules in the disease, or SPIN.

Positive predictive value (PPV) or precision is the probability that subjects with a positive screening test truly have the disease. It tracks the performance of positive example

classification. It is calculated by the number of true positive are divided by the number of condition positive (sum of false positive and true positive).

Negative predictive value (NPV) is the probability that subjects with a negative screening test truly do not have the disease. It is calculated by the number of true negative are divided by the number of condition negative (sum of false negative and true negative).

2.6 Uncertainty

Uncertainty can be defined as "not knowing for sure" as its literal meaning. The predictive uncertainty (PU) is the variance of the predictive probability. The predictive uncertainty is the sum of the aleatoric uncertainty (AU) and epistemic uncertainty (EU) [12].

$$PU = EU + AU \tag{2.5}$$

Epistemic uncertainty derives its name from the Greek word "$\varepsilon\pi\iota\sigma\tau\eta\mu\eta$" (episteme) which can be roughly translated as "knowledge". Epistemic uncertainty is calculated from the lack of knowledge of information regarding the phenomena that dictate how a system should behave, ultimately affecting the outcome of an event [13]. Epistemic uncertainty accounts for the ambiguity of the model's parameters, it can be reduced with more diverse training data. Epistemic uncertainty is modeled by placing a prior distribution over the network's weights. The epistemic uncertainty is the variability of the input data [14]. It is captured by placing a distribution over the network's weights and averaging all parameters [15].

Epistemic uncertainty can be calculated as the probability distribution over model parameters [16]. Bayesian neural network (BNN), which combines neural network with Bayesian inference, has been introduced to quantify epistemic uncertainty [17, 18, 19]. Let $D_tr = X, Y = (x_i, y_i)_i^N = 1$ be the training data set with inputs $x_i \in R^D$. $y_i \in 1, ..., C$ is a corresponding classes where C represents the number of classes. The aim is to optimize

the parameters of function $y = f^w(x)$ that can produce the desired output. The soft-max likelihood is usually used for classification problems. It is expressed as,

$$p(y = c|x, w) = \frac{exp(f_c^w(x))}{\sum_c' exp(f_c'^w(x))} \tag{2.6}$$

and the Gaussian likelihood can be assumed as,

$$p(w|X, Y) = N(y, f^w(x), \tau^{-1}I) \tag{2.7}$$

in which τ is the model precision.

The posterior $p(x|X, Y)$ is computed as

$$p(y = c|x, w) = \frac{p(Y|X, w)p(w)}{p(Y|X)} \tag{2.8}$$

The predictive distribution of the output y_o that labeled the input x_o is expressed as,

$$p(y_o|x_o, X, Y) = \int p(y_o|x_o, w)p(x|X, Y)dw \tag{2.9}$$

Aleatoric uncertainty derives its name from the Latin phrase "Ālea iacta est" which is translated as "the dice is cast". Therefore aleatoric uncertainty captures the inherent randomness in the observed data (input) and is irreducible. Aleatoric uncertainty is modeled by placing a distribution over the network's output. It is is calculated by adjusting the loss functions of the network. It captures the noise inherent in the observations [13].

Aleatoric uncertainty is modeled by placing a distribution over the output of the model [20]. For example, in regression the outputs might be modeled as corrupted with Gaussian random noise. In non-Bayesian neural networks, the observation noise parameter is fixed as part of the model's weight decay, and ignored. When the observation noise parameter is dependent on data, it can be learned as the minimization objective [21],

$$L_{NN}(\theta) = \frac{1}{N}\sum_{N}^{i=1}\frac{1}{\sigma(x_i)^2}\|y_i f(x_i)^2\| + \frac{1}{2} log\sigma(x_i)^2 \tag{2.10}$$

CHAPTER THREE

Automatic Classification of Colon Polyps

Approximately 15 million colonoscopies were undertaken in the USA in 2012 [22]. Two thirds of those colonoscopies performed in the United States are for were undertaken for diagnostic purposes. In these colonoscopies, the histologic assessment of diminutive polyps is essential to determine the type of polyps, hyperplastic or adenomatous as well as the surveillance interval. The pathology contributes a large fraction of the cost. If adenomatous polyps can be correctly distinguished from hyperplastic polyps, then they could be resected and discarded without pathologic confirmation ("diagnose and discard" strategy), whereas hyperplastic polyps could be left in situ ("diagnose and leave" strategy), significantly reducing the pathology costs of screening. In addition to that, the cost of equipment, the post-polypectomy complications, and the communication for the colonoscopy surveillance interval to the patient will also be reduced.

The American Society of Gastrointestinal Endoscopy established performance benchmarks that would allow the adoption of a diagnostic technology into practice [23]:

(1) For rectosigmoid polyps to be left in place, the technology should have a 90% negative predictive value (NPV) for adenomatous histology when used with high confidence.

(2) For polyps to be resected and discarded without pathologic assessment, the diagnostic technology should agree at least 90% of the time with the standard approach (which is based on the histologic assessment of all identified polyps) in the assignment of the post-polypectomy surveillance interval.

Colorectal cancer remains the second leading cause of cancer death for both women and men with more than 130,000 newly diagnosed cases and 50,000 deaths each year in

the United States alone. Millions of screening colonoscopies are performed in the US each year, in which the pathology costs are the majority of the total screening costs. Diminutives polyps, which are $\leq$ 5 mm in size, form 80 percentage of all found polyps. There are two types of polyps, adenomatous (progress to cancer) and hyperplastics (no cancer risk). Currently all polyps are diagnosed by the pathologist and removed by the gastroenterologist. Hence, it is an expensive procedure to get histology assessment for all polyps. If endoscopy is able to accurately distinguish between the two types of diminutive polyps, only adenomatous polyps will be removed. That can reduce the pathology costs of screening significantly.

For colonoscopy, the doctor uses a longer, thin, flexible, and light tube inserted through your colon to check for polyps or cancer inside those areas. During the test, the doctor can find and remove most polyps and some cancers. Colonoscopy also is used as a follow-up test if anything unusual is found during one of the other screening tests. It is recommend to have a colonoscopy every ten years (for people who do not have an increased risk of colorectal cancer). However if one is in high risk group, the surveillance test can be as often as every two years.

The majority of colorectal polyps found at screening colonoscopy are diminutive (5 mm or less). Diminutive polyps are either hyperplastic (non-adenomatous) polyps with a zero risk of progression to cancer or adenomatous polyps that only rarely harbor advanced histological features or cancer. Nonetheless, based on current guidelines, endoscopists are obliged to remove all diminutive polyps and submit them for histopathology. The costs of resection and pathologic examination of diminutive polyps add substantially to the total cost of colonoscopy in the United States, where an estimated 15 million screening and surveillance procedures are performed annually. Reliable endoscopic diagnosis would:

(1) obviate resection of hyperplastic polyps

(2) enable resection of adenomatous polyps without the need for histologic evaluation

This approach, which would decrease costs without affecting the rates of adenoma detection and or the surveillance recommendations, has been endorsed by the American Society of Gastrointestinal Endoscopy. Several studies have assessed computer assisted diagnosis of polyp images. In this study, we hypothesized that the automated analysis of multiple images of diminutive polyps obtained under white light imaging (WLI) and narrow band imaging (NBI) would have a high sensitivity in the diagnosis of adenomatous histology.

White light imaging (WLI) is the dominant diagnostic modality in clinical capsule endoscopy[24]. WLI limits diagnosis to the mucosal surface of the gut owing to the limited penetration depth of optical wavelengths beyond the tissue surface.

Narrow Band Imaging (NBI) is an optical imaging technology that enhances the visibility of vessels and other tissue on the mucosal surface [25]. Scopes with NBI capability illuminate the tissue with light that is strongly absorbed by hemoglobin. NBI has been widely studied to develop the real-time, endoscopic diagnosis of polyps. Scopes with NBI capability illuminate the tissue with light composed of two specific wavelengths (415 nm and 540 nm in the Olympus scopes) that are strongly absorbed by hemoglobin [26].

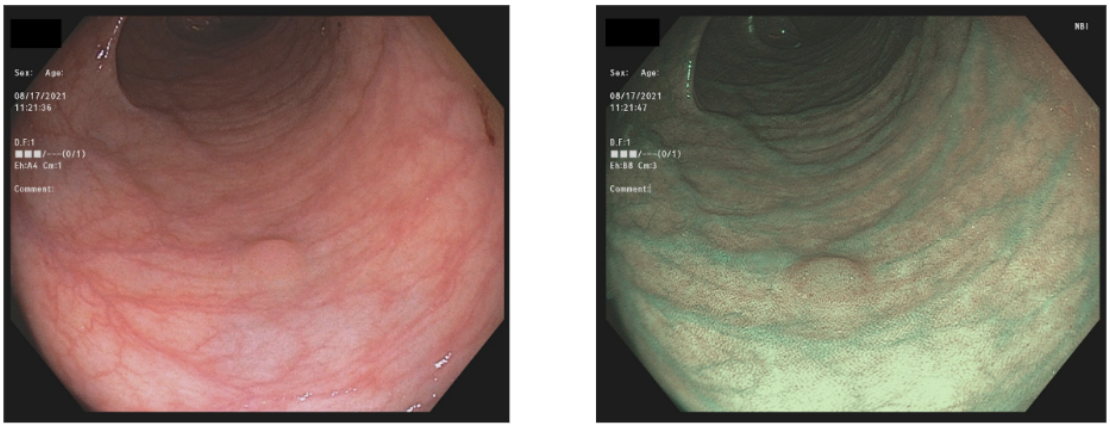

Figure 3.1: Adenoma polyp image in white light (left) and NBI (right) mode

Currently, when a patient has colon polyps, it is not treated until it enters stages 3 or 4. A camera is put up into the colon and the doctor can move it around to see if there are

any irregular protrusions in the surface. However, early stages of colon cancer are harder to find with a camera because they do not break the surface. Also, it is hard to determine at earlier stages if they are going to be cancerous or not. There are certain needs for the computer aid for early diagnosis.

Table 3.1: Summary of polyp size and its description

	Diameter	Description
Stage 0	–	No polyps. Only involves the mucosa, but nothing past the muscularis mucosa
Stage 1	$\leq$ 6 mm	Polyp has grown through muscularis mucosa and and extends into submusoca
Stage 2	6 - 9 mm	Grown through submucosa and exends through muscularis propria
Stage 3	9 - 10 mm	Grown through muscularis propria and into outmost layers of colon, but not through them
Stage 4	$\geq$ 10 mm	Grown through serosa into the outermost lining of intestines

3.1 Methodology

Figure 3.2 presents the whole study flow. There were 719 approached patients. Among them, there were 598 consented and eligible patients. There were 346 patients with recorded polyps with in those eligible and consented patients. The database has 514 polyps.

The study consists of two main parts: clinical images gathering and deep learning predicting.

(1) Clinical images gathering happens at the Baylor University Medical Center (BUMC) at Dallas and surrounding clinics. Each case consists of:

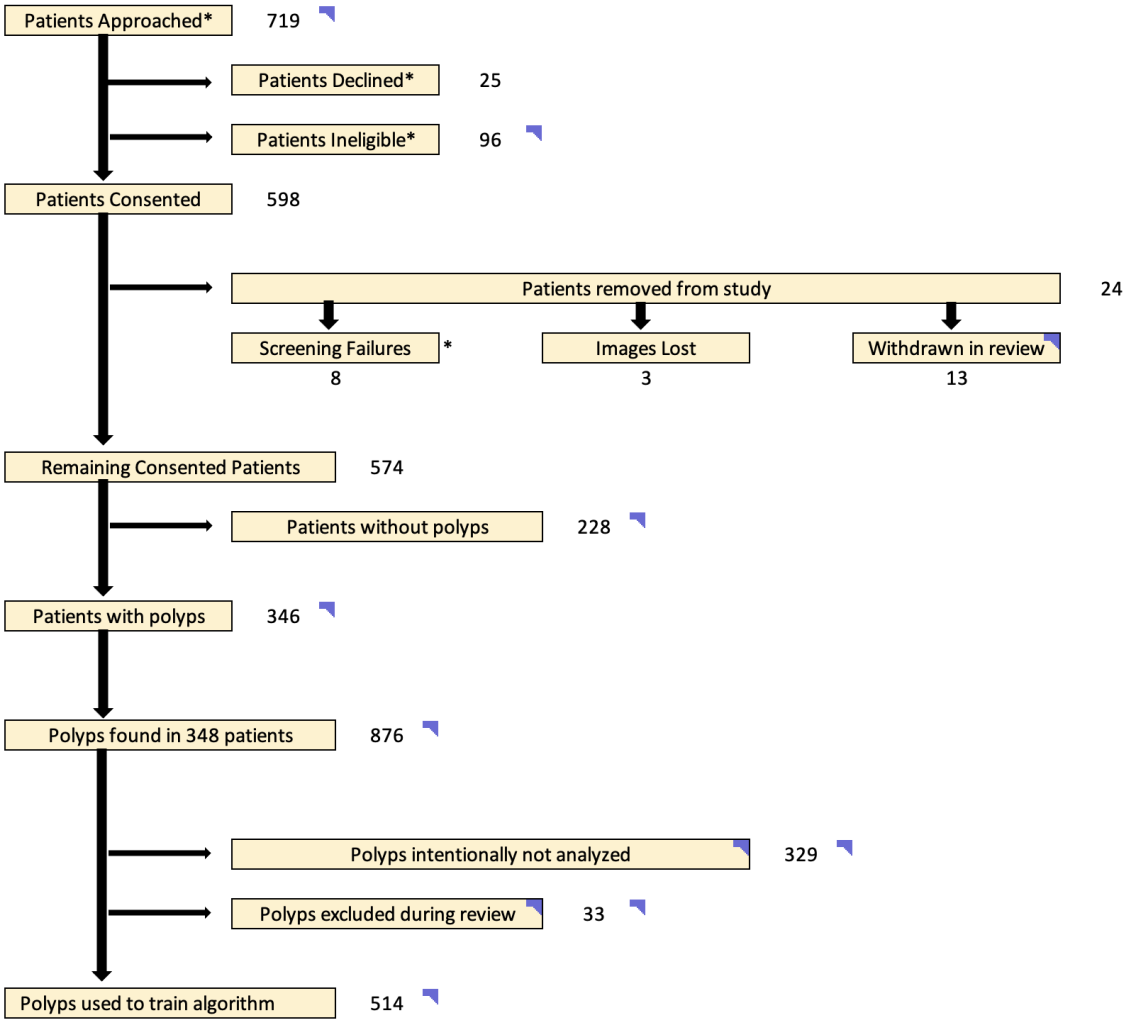

Figure 3.2: Study Flow of the Study

- patient demographics (age, gender, race, and ethnicity) and family history of colorectal cancer
- WL and NB images of the diminutive polyp
- endoscopic data (colonoscopy indication, polyp size and location, and endoscopist's optical diagnosis)
- pathological diagnosis

(2) Deep learning predicting is done at Baylor University in Waco. The anonymous clinical data and images are transferred via ProofPoint Secure.

- Data prepossessing
- CNNs consist of many layers which includes convolutional layer, pooling layer and fully connected layer. Testing was done using a 10-fold cross validation. The image data was randomly split into 10 groups ensuring that all images of a polyp stayed in the same group.
    * Polyp detection
    * Polyp size estimation
    * Polyp histology prediction

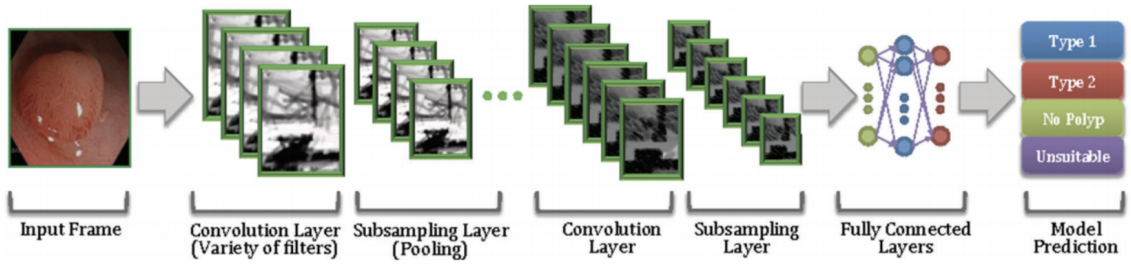

Figure 3.3: Overview of the training flow in polyp project

3.2 Clinical Data Collection

A collaboration of endoscopists at Baylor University Medical Center at Dallas (BUMC) and Baylor Scott and White (BSW)-Temple and electrical computer engineers at Baylor University (BU) was formed to work on this project. Eligible subjects were adults (age 18 or older) who underwent a screening or surveillance colonoscopy at BUMC and at BSW-Temple. The research assistants screened the endoscopic schedule and approached eligible individuals. Individuals who consented to the study were assigned a unique identifying number. For each subject, we recorded the procedure indication (screening or surveillance), age, gender, race, and ethnicity.

We used a video processor (EVIS EXERA III) and colonoscope (CF-HQ190L/I, PCF-H190L/I) manufactured by Olympus (Olympus America, Center Valley, PA). The

video processor has a dual focus that enables with HQ endoscopes. It allows the user to switch between two focus settings with the push of a button. ”Near mode” features ground-breaking resolving power for close mucosal observation and “far mode” delivers normal observation. EVIS EXERA III endoscopes feature an advanced level of resolving power compared to previous generations. NBI in these endoscopes offers up to twice the viewable distance in the lumen compared to previous generations. A range of signal output options are available to suit the monitor attached to the CV-190. This enables achieving the optimal image for the installed infrastructure. 16:9 and 16:10 format is available for HD monitors.

The endoscopists were asked to image at least one polyp for the purposes of the study. Imaging of any additional polyp was at the discretion of the endoscopist, who had to consider potential procedure delays due to image acquisition. The endoscopists were asked to obtain focused, close-up images of polyps 5 mm or less, from different angles, while minimizing reflections, mucus and fecal debris. Per the initial protocol, we acquired at least 3 still images under WLI and 3 still images under NBI. Images were captured using the MediCap USB300 High Definition Medical Video Recorder (MediCapture, Plymouth Meeting, PA). The endoscopist determined the location of each polyp, estimated its size by using a forceps for reference and captured an image of the polyp and the adjacent forceps. Aiming to potentially improve the diagnostic characteristics of our technology, we subsequently revised the protocol to obtain 5 still images under WLI setting and 5 still images under NBI setting.

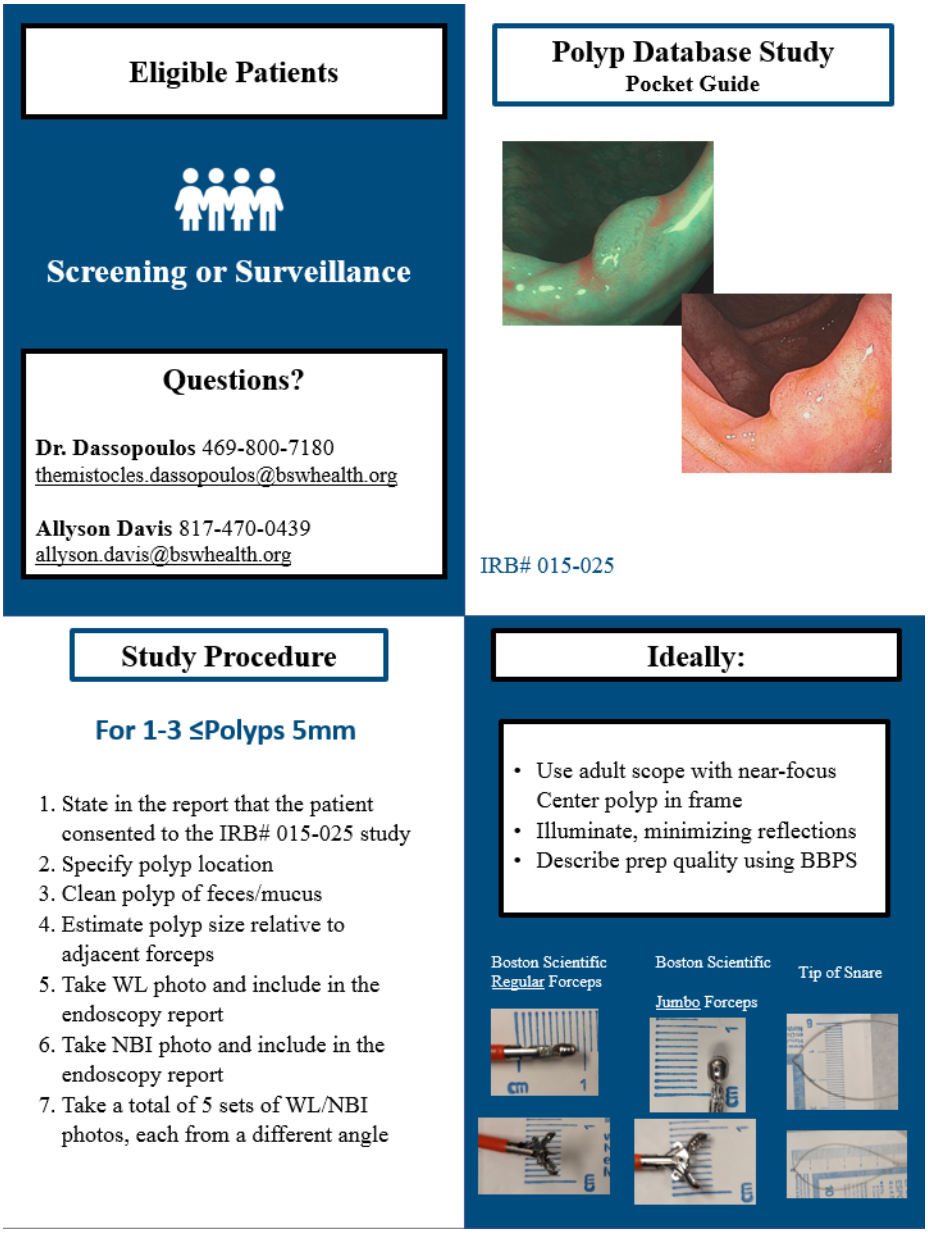

Figure 3.4: Pocket guide of Polyp Study

Figure 3.4 is a pocket guide of polyp study that was designed and gave to the prospective patients as well as any participating physician in the study. The study procedure was explained above with the contact information of the principle investigator of this project at BSW-Dallas. There are examples of forceps and snares used in the study. There are two biopsy devices commonly used in the procedure. A forceps is a device used to either biopsy or completely remove small polyps. A snare is a device with a wire loop, which can be wrapped around the base of a polyp. In this study, a picture of a polyp that next to a forceps or snare was taken to be a preference picture of the polyp estimated size. Due to the nature of taking images in various clinics, Boston Scientific and Cook Medical are two brands of forceps used to estimate the size of polyps. Within each brand, there are regular and jumbo forceps, as well of a snare.

3.3 Result

A database of clinical polyps in WLI and NBI modes was built for the purpose of this study. The images in that database were used to investigate the method of predicting polyp histology with CNNs. The accuracy, sensitivity, and specificity results are greater than 97%. Most importantly the technique has a 95.48% negative predictive value, exceeding the 90% value set by ASGE. The 95% confidence intervals were calculated. If a series of samples are drawn and the mean of each calculated, 95% of the means would be expected to fall within the range of two standard errors above and two below the mean of these means. This common mean would be expected to lie very close to the mean of the population. So the standard error of a mean provides a statement of probability about the difference between the mean of the population and the mean of the sample.

The probability of having the disease (positive), given the results of a classification, is called the predictive value of the test. Positive predictive value is the probability that a polyp with a positive (adenoma) histology result actually is adenomatous. Negative predictive value is the probability that a polyp with a negative (hyperplastic) histology result is truly free of disease.

3.3.1 The Whole Data Set

Figure 3.2 shows the overall result of the classification of polyps in the whole data set, in WLI mode only and in NBI mode only. In this table, ALL represents the results of the first test in which images both light modes were used in training and testing. WLI column shows the result of the second test in which only images in WLI mode were used in training and testing. NBI column gives the results of the third test in which only images in NBI mode were used in training and test. These tests were performed to show how good the model is when using all images that collected versus each individual imaging mode. It is also used to justified the protocol of the study - to use images from both image modes instead of just NBI.

Table 3.2: GI result all

	ALL	95% CI	WL	95% CI	NIB	95% CI
Total polyps	495		495		495	
Accuracy	96.16%		94.14%		95.15%	
Sensitivity	96.36%	94.15% - 98.58%	93.82%	90.97% - 96.67%	95.27%	92.76% - 97.78%
Specificity	95.91%	93.29% - 98.53%	94.55%	91.55% - 97.55%	95.00%	92.13% - 97.88%
Positive Predictive Value	96.72%	94.61% - 98.87%	95.56%	93.10% - 98.04%	95.97%	93.64% - 98.83%
Negative Predictive Value	95.48%	92.74% - 98.22%	92.44%	88.99% - 95.90%	94.14%	91.06% - 97.23%

The accuracy of the model for all polyps in the ratio of the correct predictions to the total number of polyps in the data set. The overall accuracy is 96.16%. The accuracy of WLI setting and NBI setting are 94.14% and 95.15% respectively. The different in

the results is about 1%, and the overall using both image modes got the best accuracy result. The accuracy shows that using both WL and NB images improve the result of the classification. In the not ideal situation in which only one mode of lightning is present, the model is still good enough to classify the type of polyps. However the overall result improved when both modes were used. This is predictable and expected when the protocol was designed. This proved that the approach is in the right direction.

The sensitivity and the specificity are above 93% for all three cases. The sensitivity is important because it shows that NPV are 95.48%, 92.44%, and 94.14% for the overall data set, WLI setting only and NBI setting only. The WLI only set performed the worst out of the three sets.

Figure 3.5 is the confusion matrix of the overall data set. The number of FN and FP polyps in the overall data set is 19 while it is 29 and 24 in WLI and NBI only modes as in figure 3.7 and figure 3.6 respectively. Within those 29 and 24 polyps are the same 18 (out of 19) polyps from the whole data set.

From the metrics in the confusion matrices, it shows that if the model has accessed to both WL and NB images, it performed better and gave more accurate classification. Even though the 1% increment seems small, it is in the best interest of the physicians to archive the most accurate result in diagnostics, with the highest degree of confidence.

		Condition		
		Gold Standard:	Pathologist	
	Total Population TotPop=CP+CN =TestP+TestN 495	Condition Positive CP = TP + FN 275	Condition Negative CN = TN + FP 220	Prevalence Prev = CP/TotPop 0.555555556	
Test Outcome	Test Positive TestP = TP + FP 274	True Positive TP 265	False Positive Type 1 Error FP 9	Positive Predictive Value, Precision PPV = TP/TestP 0.967153285	False Discovery Rate FDR = FP/TestP 0.032846715
	Test Negative TestN = TN + FN 221	False Negative Type 2 Error FN 10	True Negative TN 211	False Omisson Rate FOR = FN/TestN 0.045248869	Negative Predictive Value NPV = TN / TestN 0.954751131
	Accuracy ACC = (TP+TN)/TotPop 0.96161616	Sensitivity True Positive Rate TPR = TP /CP 0.963636364	Fall-out False Positive Rate FPR = FP/CN 0.040909091	Positive Likelihood Ratio LR+ = TPR/FPR 23.55555556	Diagnostic Odds Ratio DOR = LR+/LR- 0.893101632
		Miss Rate False Negative Rate FNR = FN /CP 0.036363636	Specificity True Negative Rate TNR = TN/CN 0.959090909	Negative Likelihood Ratio LR- = TNR/FNR 26.375	

Figure 3.5: GI confusion matrix for all polyps

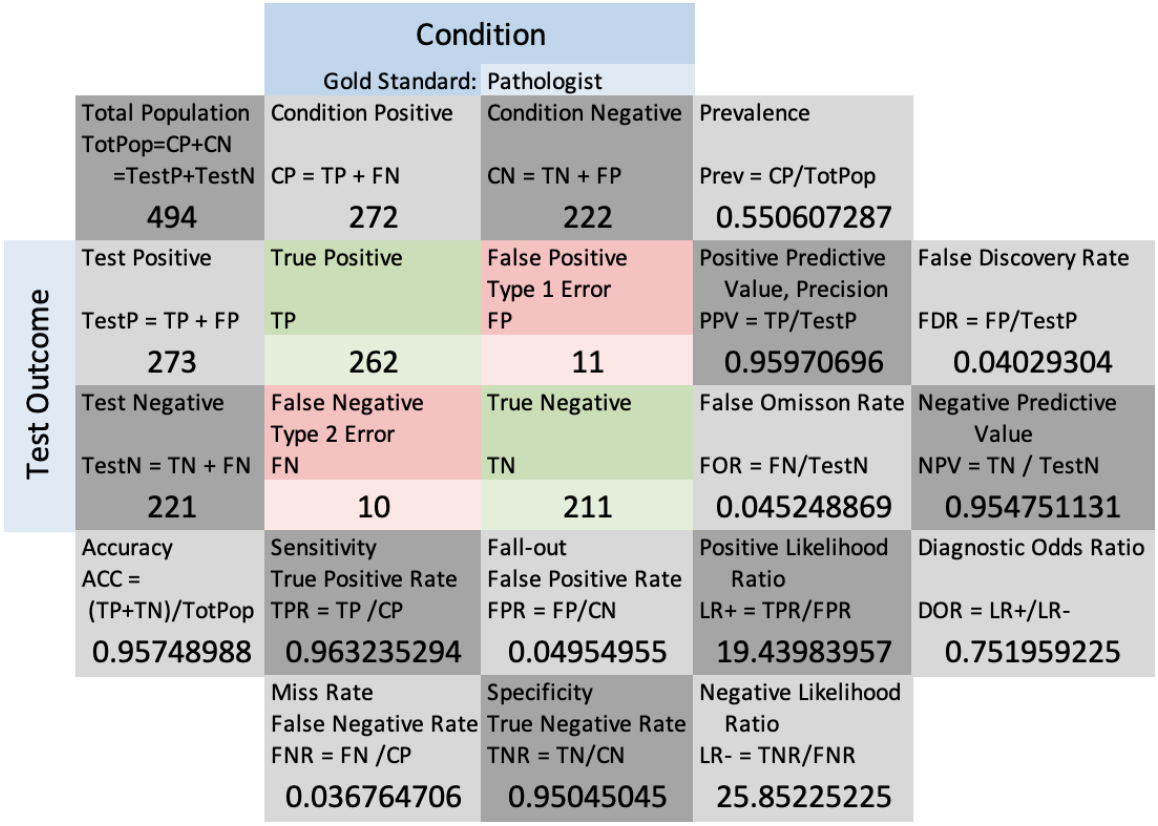

Figure 3.6: GI confusion matrix for all polyps in NBI only

		Condition			
		Gold Standard:	Pathologist		
	Total Population TotPop=CP+CN =TestP+TestN 491	Condition Positive CP = TP + FN 268	Condition Negative CN = TN + FP 223	Prevalence Prev = CP/TotPop 0.545824847	
Test Outcome	Test Positive TestP = TP + FP 270	True Positive TP 258	False Positive Type 1 Error FP 12	Positive Predictive Value, Precision PPV = TP/TestP 0.955555556	False Discovery Rate FDR = FP/TestP 0.044444444
	Test Negative TestN = TN + FN 221	False Negative Type 2 Error FN 10	True Negative TN 211	False Omisson Rate FOR = FN/TestN 0.045248869	Negative Predictive Value NPV = TN / TestN 0.954751131
	Accuracy ACC = (TP+TN)/TotPop 0.95519348	Sensitivity True Positive Rate TPR = TP /CP 0.962686567	Fall-out False Positive Rate FPR = FP/CN 0.053811659	Positive Likelihood Ratio LR+ = TPR/FPR 17.88992537	Diagnostic Odds Ratio DOR = LR+/LR- 0.705498578
		Miss Rate False Negative Rate FNR = FN /CP 0.037313433	Specificity True Negative Rate TNR = TN/CN 0.946188341	Negative Likelihood Ratio LR- = TNR/FNR 25.35784753	

Figure 3.7: GI confusion matrix for all polyps in WL only

3.3.2 Gender

In this section, the result of classifying polyps in male versus female participants will be presented. From statistics, the chance of male population who has colon cancer is higher than the female population [27]. However, there are more female patients who agreed to participate into the study than male patients. This section will show that by comparing the result of polyps in male versus polyps in female in this study, it will determine what type of data should be gathered to strengthen the methodology.

Figure 3.8 and Figure 3.9 demonstrate the confusion matrices of the overall data set (using both imaging modes) for male and female patients. There were 340 total patients in which there were 208 female and 132 male. 299 polyps were collected from female patients and 196 polyps were collected from male patients. The accuracy for the population of male only is 95.91% while it is 96.32% for female only. The difference between the results is less than 0.5%, which is small enough to say that the gender of patient is not the deciding factor for the result. It is expected but showed in the number. The result from male only polyps has higher sensitivity at 97.43% when from female only polyps has 95.59%. NPV in male only polyps is at 96.10% while from female only polyps is at 95.10%. Male only polyps results perform better in the sensitivity and NPV while is about 0.5% less accurate than the female only results. 0.5% is small but considering the male only polyps total number is 34% (or 97 polyps) less than the female only population, if there are more polyps from male population, the result is believed to be improved and increased in accuracy.

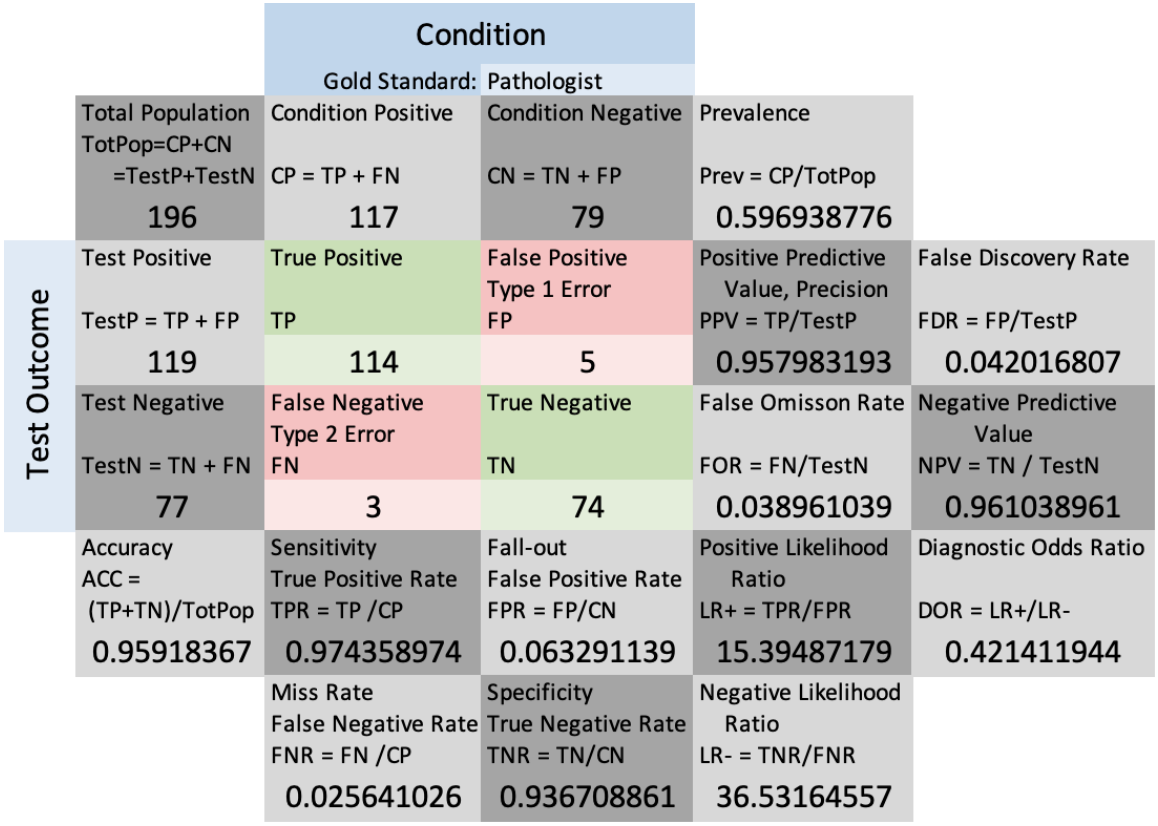

Figure 3.8: GI confusion matrix for all polyps in Male only

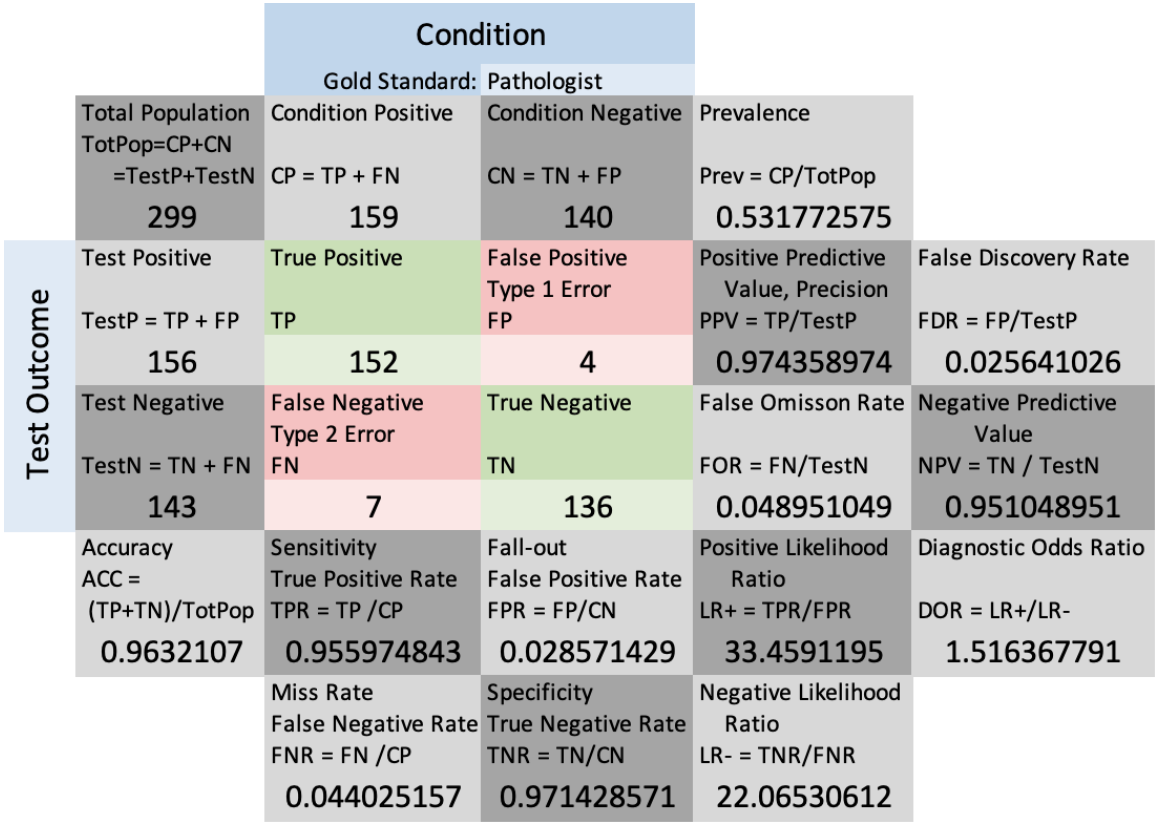

Figure 3.9: GI confusion matrix for all polyps in Female only

To further investigate in this matter, testing and training using specific gender polyps were demonstrated. Figure 3.10 and Figure 3.11 demonstrate the confusion matrices of the results when only images from one gender were used in training and testing. The numbers show that the results has lower accuracy, sensitivity and NPV than when training and testing used the whole data set.

		Condition			
		Gold Standard: Pathologist			
	Total Population TotPop=CP+CN =TestP+TestN 196	Condition Positive CP = TP + FN 117	Condition Negative CN = TN + FP 79	Prevalence Prev = CP/TotPop 0.596938776	
Test Outcome	Test Positive TestP = TP + FP 116	True Positive TP 110	False Positive Type 1 Error FP 6	Positive Predictive Value, Precision PPV = TP/TestP 0.948275862	False Discovery Rate FDR = FP/TestP 0.051724138
	Test Negative TestN = TN + FN 80	False Negative Type 2 Error FN 7	True Negative TN 73	False Omisson Rate FOR = FN/TestN 0.0875	Negative Predictive Value NPV = TN / TestN 0.9125
	Accuracy ACC = (TP+TN)/TotPop 0.93367347	Sensitivity True Positive Rate TPR = TP /CP 0.94017094	Fall-out False Positive Rate FPR = FP/CN 0.075949367	Positive Likelihood Ratio LR+ = TPR/FPR 12.37891738	Diagnostic Odds Ratio DOR = LR+/LR- 0.801491782
		Miss Rate False Negative Rate FNR = FN /CP 0.05982906	Specificity True Negative Rate TNR = TN/CN 0.924050633	Negative Likelihood Ratio LR- = TNR/FNR 15.44484629	

Figure 3.10: GI confusion matrix for all polyps in Male only, training and testing using Male only polpys

		Condition			
		Gold Standard:	Pathologist		
	Total Population TotPop=CP+CN =TestP+TestN 299	Condition Positive CP = TP + FN 159	Condition Negative CN = TN + FP 140	Prevalence Prev = CP/TotPop 0.531772575	
Test Outcome	Test Positive TestP = TP + FP 153	True Positive TP 150	False Positive Type 1 Error FP 3	Positive Predictive Value, Precision PPV = TP/TestP 0.980392157	False Discovery Rate FDR = FP/TestP 0.019607843
	Test Negative TestN = TN + FN 146	False Negative Type 2 Error FN 9	True Negative TN 137	False Omisson Rate FOR = FN/TestN 0.061643836	Negative Predictive Value NPV = TN / TestN 0.938356164
	Accuracy ACC = (TP+TN)/TotPop 0.95986622	Sensitivity True Positive Rate TPR = TP /CP 0.943396226	Fall-out False Positive Rate FPR = FP/CN 0.021428571	Positive Likelihood Ratio LR+ = TPR/FPR 44.02515723	Diagnostic Odds Ratio DOR = LR+/LR- 2.546559157
		Miss Rate False Negative Rate FNR = FN /CP 0.056603774	Specificity True Negative Rate TNR = TN/CN 0.978571429	Negative Likelihood Ratio LR- = TNR/FNR 17.28809524	

Figure 3.11: GI confusion matrix for all polyps in Female only, training and testing using Female only polpys

3.3.3 Ethnicity

In this section, the result of classifying polyps in different race and ethnicity will be presented. From statistics, in the United States, the population has the higher chance of colon rectal cancer is African American. African Americans are about 20% more likely to get colorectal cancer and about 40% more likely to die from it than most other groups[28]. From the race breakdown, there are 340 total patients, in which there were 243 Caucasian, 87 African American, 6 Asian, and 4 declined to answer. This section will show that by comparing the result of individual race in this study, it will determine what type of data should be gathered to strengthen the methodology. It will answer the question if there is a need to target and collect more polyps from once specific race versus collecting every possible polyps.

Only the results from Caucasian and African American are presented here due to the limited number of other races which are less than 6 patients for each other race, and less than 10 polyps collected in those population. It is important to acknowledge that it would be ideal if there is the equal amount of polyps in each group. Since that is not the case, investigating each individual case which presentable results is the next best thing.

Figure 3.12 and Figure 3.13 demonstrate the confusion matrices of the overall data set (using both imaging modes) for Caucasian and African American patients. 329 polyps were collected from Caucasian patients and 106 polyps were collected from African American patients. The accuracy for the population of Caucasian patients only is 95.44% while it is 93.51% for African American only. The difference between the results is about 1.6%. The result from Caucasian patients only polyps has higher sensitivity at 94.49% when from African American only polyps has 92.31%. NPV in Caucasian patients only polyps is at 97.23% while from African American only polyps is at 95.59%. Caucasian patients only polyps results perform better in accuracy and the sensitivity than the African American only results. 1.6% seems to be small but considering the Caucasian patients only polyps total number is 68% (or 223 polyps) higher than the African American only population, if there are more polyps from African American only population, the performance can be improved.

		Condition			
		Gold Standard:	Pathologist		
	Total Population TotPop=CP+CN =TestP+TestN 329	Condition Positive CP = TP + FN 109	Condition Negative CN = TN + FP 220	Prevalence Prev = CP/TotPop 0.331306991	
Test Outcome	Test Positive TestP = TP + FP 112	True Positive TP 103	False Positive Type 1 Error FP 9	Positive Predictive Value, Precision PPV = TP/TestP 0.919642857	False Discovery Rate FDR = FP/TestP 0.080357143
	Test Negative TestN = TN + FN 217	False Negative Type 2 Error FN 6	True Negative TN 211	False Omisson Rate FOR = FN/TestN 0.02764977	Negative Predictive Value NPV = TN / TestN 0.97235023
	Accuracy ACC = (TP+TN)/TotPop 0.95440729	Sensitivity True Positive Rate TPR = TP /CP 0.944954128	Fall-out False Positive Rate FPR = FP/CN 0.040909091	Positive Likelihood Ratio LR+ = TPR/FPR 23.0988787	Diagnostic Odds Ratio DOR = LR+/LR- 1.325732418
		Miss Rate False Negative Rate FNR = FN /CP 0.055045872	Specificity True Negative Rate TNR = TN/CN 0.959090909	Negative Likelihood Ratio LR- = TNR/FNR 17.42348485	

Figure 3.12: GI confusion matrix for all polyps in Caucasian patients only

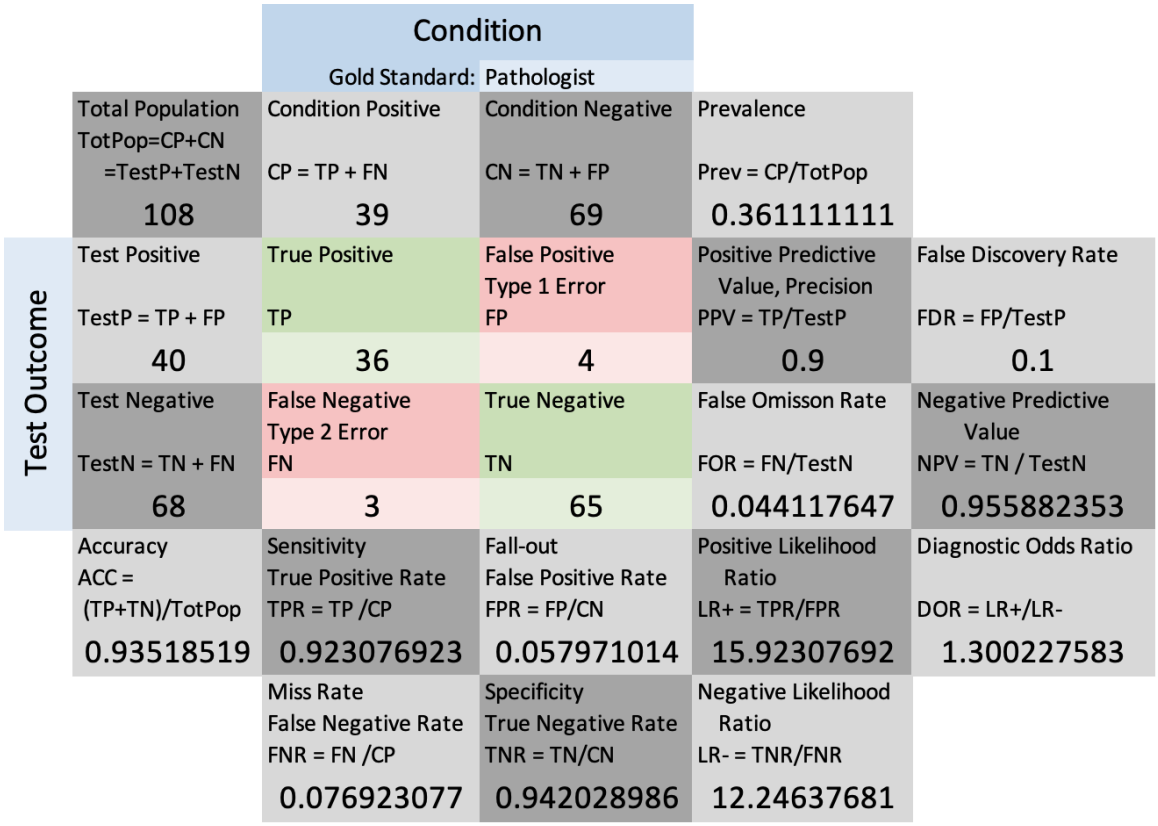

Figure 3.13: GI confusion matrix for all polyps in African American patients only

3.3.4 Location

In this section, the results from different part of the colon are presented. The large intestine includes the cecum, appendix, entire colon, rectum, and anal canal. The colon is the longest segment of the large intestine. The colon extends from the cecum up the right side of the abdomen, across to the left side, down the left side and finally loops at the sigmoid colon to join the rectum. The colon parts between the cecum and sigmoid colon are the ascending colon), transverse colon, and descending colon. The colon absorbs the remaining fluids and salts, lubricates and stores waste products until those waste are ready to be out of the body. The ascending and transverse colon are where most absorption occurs. The liquid material received from the small intestine is dehydrated to form a fecal mass [29].

The small intestine absorbed nutrients. The large intestine absorbed water. They are composed of different tissues. The inner wall of the colon consists of a mucous membrane that absorbs the fluids and secretes mucus to lubricate the waste materials. The deeper muscle layer is composed of circular and longitudinal muscles. Circular muscles produce the mild churning and mixing motions of the intestine, while the longitudinal ones create the strong massive muscle contractions that actually move the waste [30]. Since they are composed of different types of tissues at different section of the colon for different purpose, the process to differentiate those tissues in the classification task is interesting to investigate.

The way that polyps were collected in the protocol is that when a clean polyps was seen (by the physician), its location and estimated size (relative to adjacent forceps) were recorded. Then a WL and NB image were taken at one angle, then repeated another four times. To not disturb the procedure, a maximum of 5 polyps per patient were taken starting with the ascending colon to the transverse colon. As a result the location of the polyps were different in different patients resulting in regional tissue difference. Recent studies [31] have shown regional difference in cancer propensities and lethalities between men and women. Testing polyp recognition in each area of the colon is thus essential.

The difference is from the different patients or the different location of polyps which have different structures/tissues - regional difference. The breakdown in details of the results can be used to investigated.

Proximal to the sigmoid colon is the parts of the colon above or before the sigmoid colon. It includes cecum, ileocecal valve, ascending colon, hepatic flexure, transverse colon, splenic flexure, and descending colon. Figure 3.14 shows the classified result for the polyps in proximal to the sigmoid colon. There are 339 polyps in this set and both of WL and NB images were used for each polyp. The accuracy for this set of polyps is 96.76%.

Rectosigmoid colon includes sigmoid colon and rectum. Figure 3.15 shows the classified result for the polyps in rectosigmoid colon. There are 156 polyps in this set and both of WL and NB images were used for each polyp. The accuracy for this set of polyps is 94.87%. For the rectosigmoid polyps if there are 197 (72 adenomatous and 125 non-adenomatous) polyps added to the database to the current 162 rectosigmoid polyps (of which 36 adenomatous and 126 non-adenomatous), the PPV will improve from 86.84% (95% CI 76.09% - 97.59%) to 90.83% (95% CI 85.41% - 96.24%). The sensitivity 95% CI will also be smaller, from 95% CI 82.64% - 100.00% to 95% CI 86.45% - 96.88%.

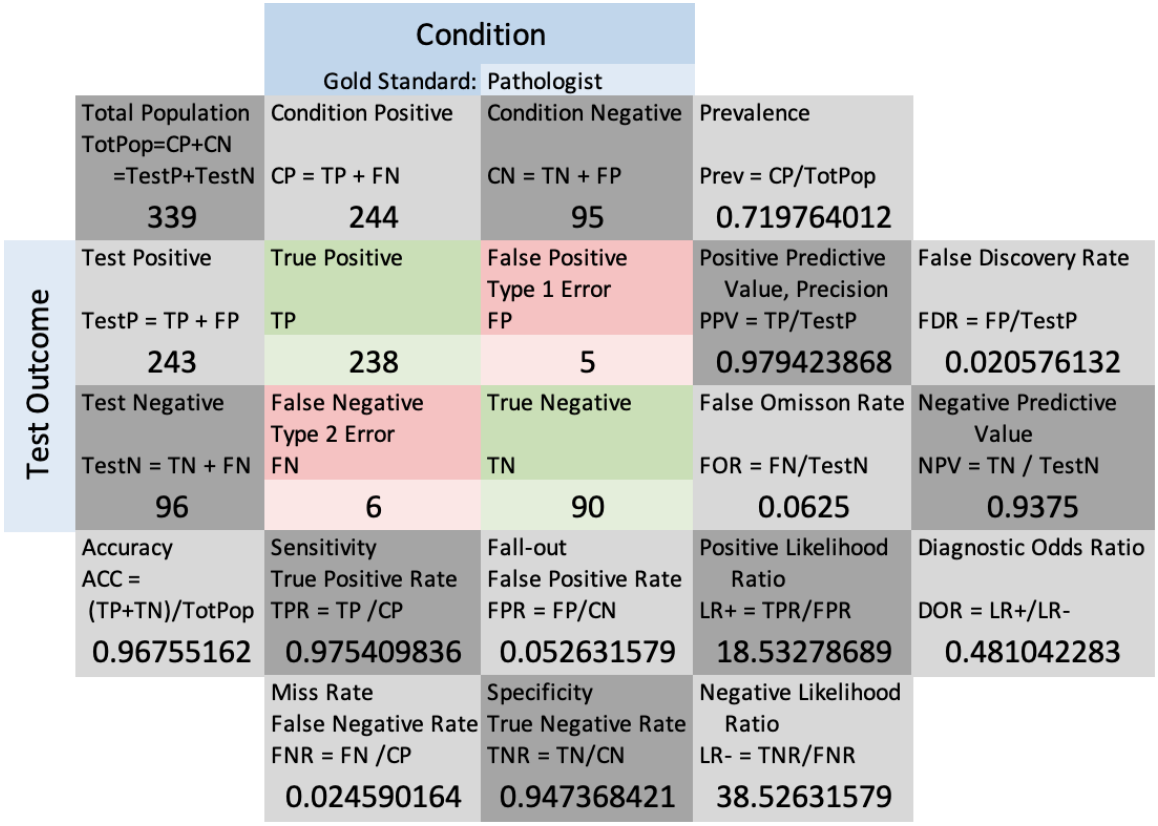

Figure 3.14: GI confusion matrix for Proximal to the sigmoid colon polyps

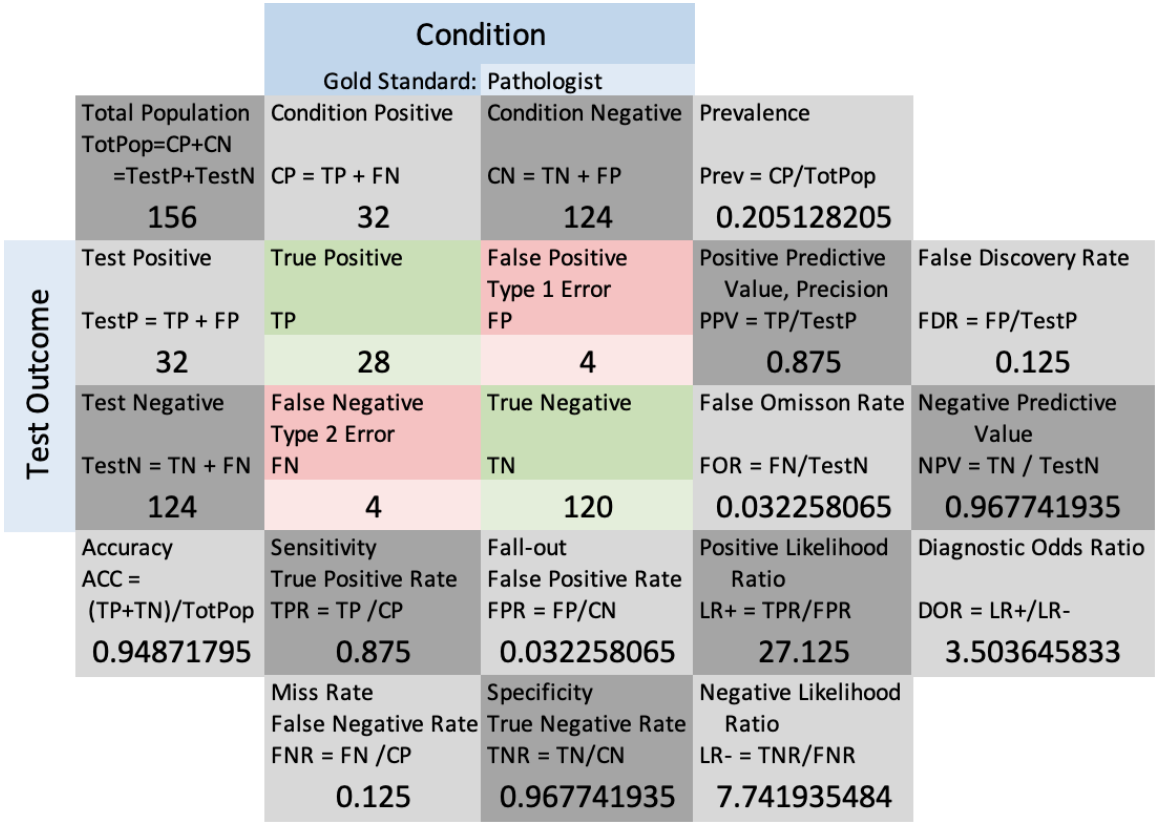

Figure 3.15: GI confusion matrix for Rectosigmoid colon polyps

3.3.5 Size Breakdown

Figure 3.3 shows the overall result of the classification of polyps in size - range from 1-2mm, 2-3mm, 3-4mm, and 4-5mm or the whole data set, in WLI and NBI settings. In this table, each size range result is shown with the corresponding 95% confidence interval. These tests were performed to show how good the model is when classifying different size of polyps. The size here is estimated and recorded by the physician during the procedure.

This section of the result is essential because polyp size is a critical bio-marker that correlates with its risk of malignancy and guides for clinical management. According to the adenoma-carcinoma theory, adenomatous polyps are the precursors of most colorectal cancers [32]. Cancerous polyps tend to grow slowly. It is estimated that the polyp dwell time, the time needed for a small adenoma to transform into a cancer, may be on average 10 years [32]. Therefore, if the sign of cancer is detected when the polyp is as small as possible, the better it will be to find out and suggest appropriate treatment or surveillance steps.

Figure 3.3 shows the overall result of the classification of polyps that broken down into size. The accuracy of the model for polyps in size 1mm to 2 mm is 97.92%. The accuracy of model for polyps in size 2 mm to 3 mm is 96.189%. The accuracy of model for polyps in size 3mm to 4mm is 92.41%. The accuracy of model for polyps in size 4mm to 5mm is 96.23%. The accuracy shows that all accuracy is above 92%. The sensitivity and the specificity are above 95% for all four cases. NPV are 98.04%, 97.65%, 92.31%, and 91.38% for the size 1 mm to 2 mm, 2 mm to 3 mm, 3 mm to 4 mm, and 4 mm to 5 mm.

Figure 3.16 shows the confusion matrix result of the classification of polyps that are 1 mm to 2 mm in size. There are 96 polyps in this set. The accuracy of the model for polyps in size 1mm to 2 mm is 97.92%. The sensitivity is 97.78% while the specificity is 98.04%. NPV is 98.04%.

Figure 3.17 shows the confusion matrix result of the classification of polyps that are 2 mm to 3 mm in size. There are 161 polyps in this set. The accuracy of the model for

Table 3.3: GI result size

	1-2 mm	95% CI	2-3 mm	95% CI	3-4 mm	95% CI	4-5 mm	95% CI
Total polyps	96		161		79		159	
Accuracy	97.92%		96.89%		92.41%		96.23%	
Sensitivity	97.78%	93.47% - 102%	97.33%	93.69% - 101%	96.08%	90.75% - 101%	95.24%	91.16% - 99.31%
Specificity	98.04%	94.23% - 101%	96.51%	92.63% - 100%	85.71%	72.75% - 98.68%	98.15%	94.55% - 102%
Positive Predictive Value	97.78%	93.47% - 102%	96.05%	91.68% - 100%	92.45%	85.34% - 99.56%	99.01%	97.08% - 101%
Negative Predictive Value	98.04%	94.23% - 102%	97.65%	94.43% - 101%	92.31%	82.06% - 102%	91.38%	84.16% - 98.60%

these polyps is 96.89%. The sensitivity is 97.33% while the specificity is 96.51%. NPV is 97.65%.

Figure 3.18 shows the confusion matrix result of the classification of polyps that are 3 mm to 4 mm in size. There are 79 polyps in this set. The accuracy of the model for polyps in this set is 92.41%. The sensitivity is 96.08% while the specificity is 85.71%. NPV is 92.31%. This polyps set had the worst performance among all size group. It is noticed that there is an imbalance in the data set, only 51 polyps in the positive set and 28 polyps in the negative set. In this size range, even though there are only 6 missed classified cases, it affected the overall results a lot.

Figure 3.19 shows the confusion matrix result of the classification of polyps that are 4 mm to 5 mm in size. There are 159 polyps in this set. The accuracy of the model for these polyps is 96.23%. The sensitivity is 95.24% while the specificity is 98.15%. NPV is 91.38%.

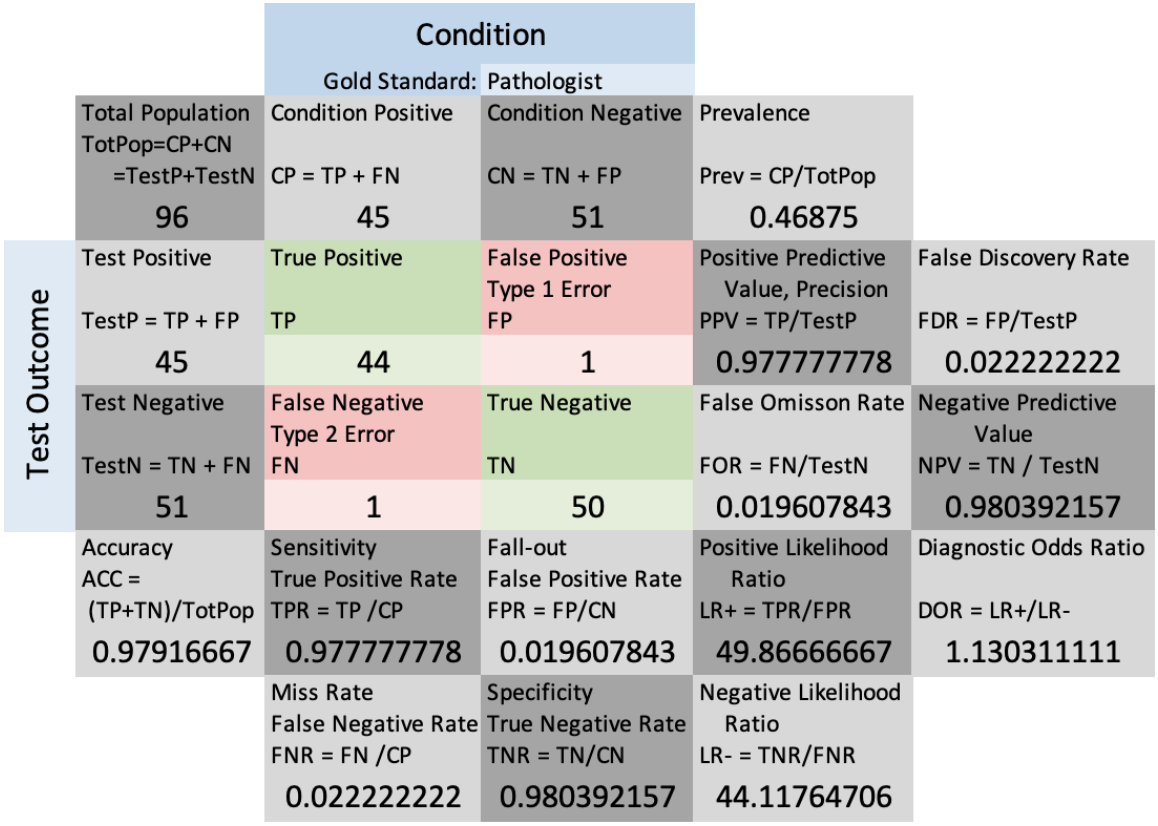

Figure 3.16: GI confusion matrix for 1 mm to 2 mm in diameter polyps

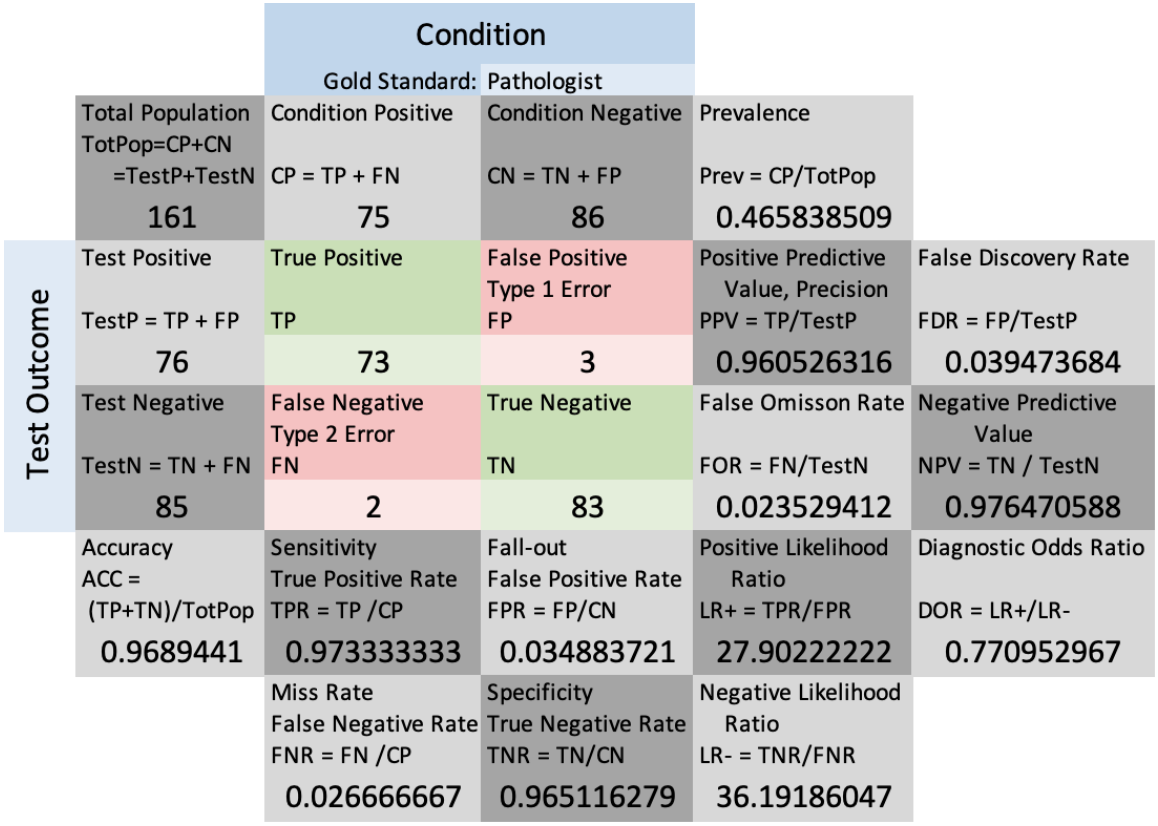

Figure 3.17: GI confusion matrix for 2 mm to 3 mm in diameter colon polyps

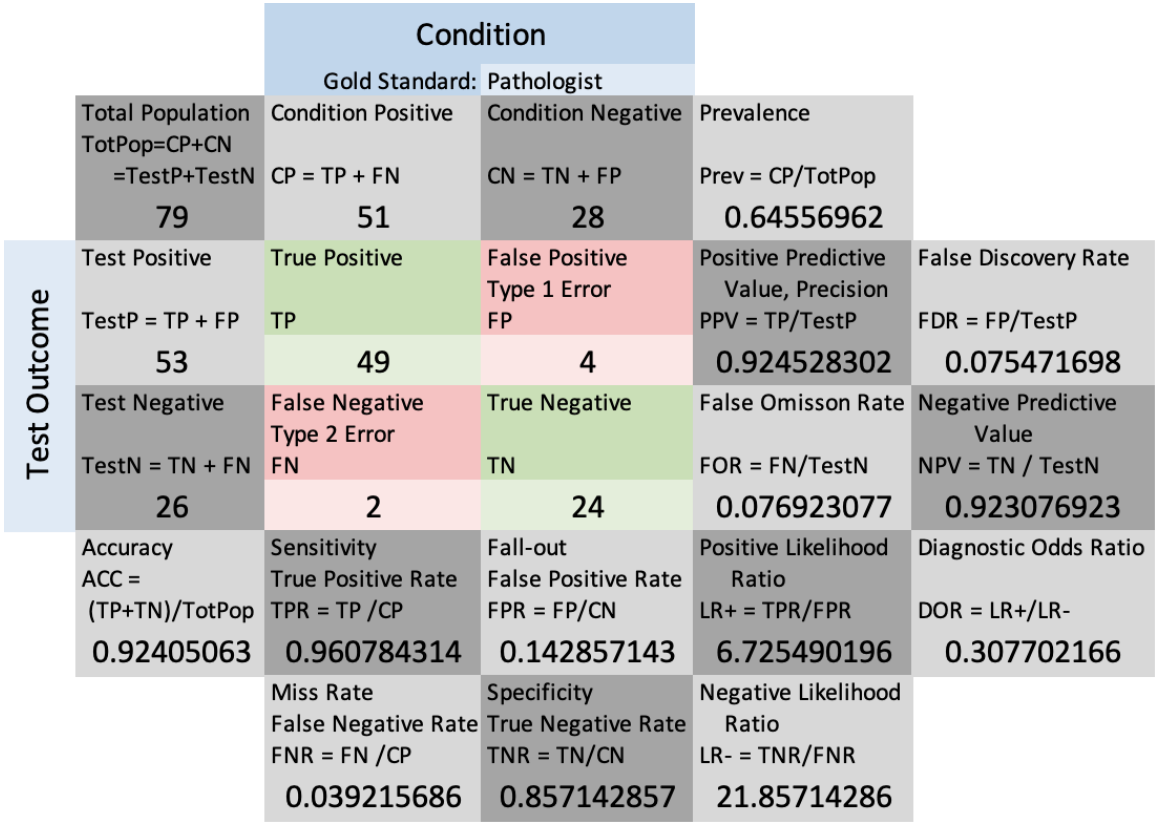

Figure 3.18: GI confusion matrix for 3 mm to 4 mm in diameter colon polyps

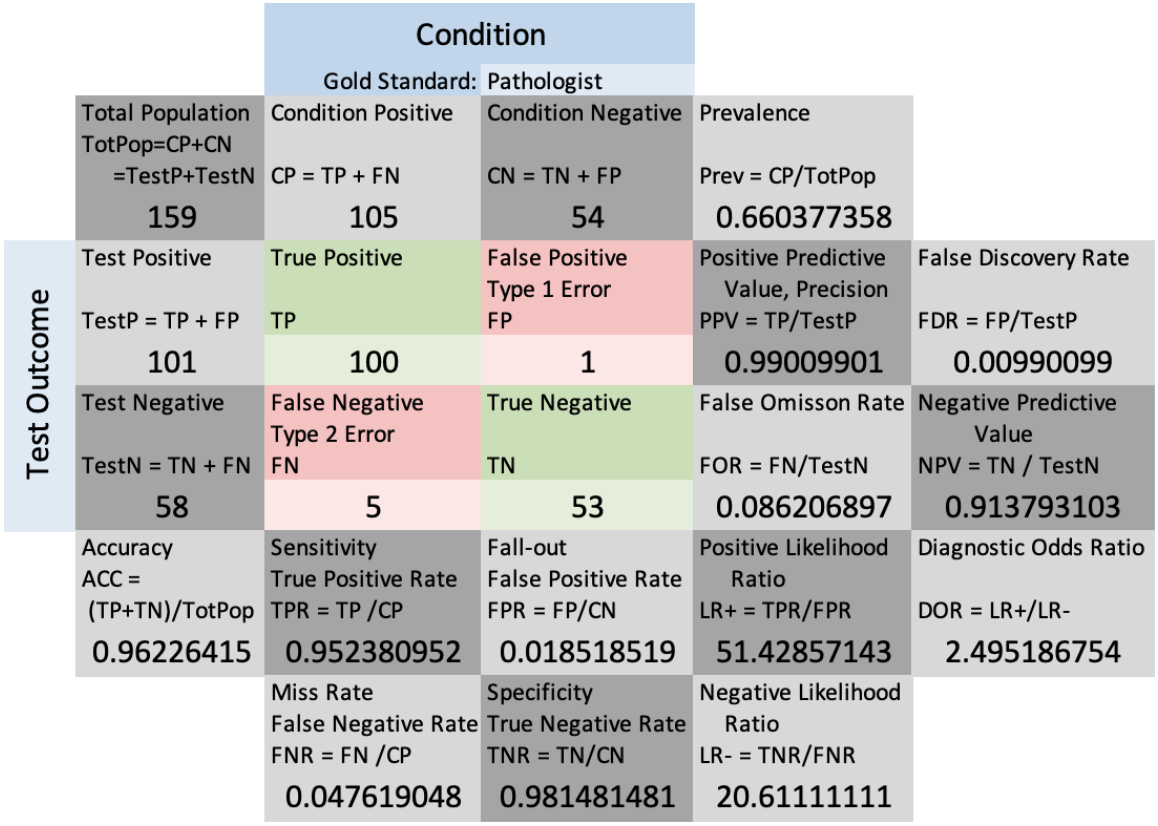

Figure 3.19: GI confusion matrix for 4 mm to 5 mm in diameter colon polyps

3.3.6 Age Group

In this section, the result for different age group is presented. According in a recent study in 2021, the American Cancer Society's newest guidelines recommend that colorectal cancer screenings begin at age 45 for average risk patients because colorectal cancer cases are on the rise among young and middle-age people. According to the National Cancer Institute website in 2022, deaths of people under age 55 increased 1% per year from 2008 to 2017, even though overall colorectal cancer rates have dropped. The age for colonoscoy screening should be lower for high risk patients who are either in these following catergories:

- A strong family history of colorectal cancer or certain types of polyps
- A personal history of colorectal cancer or certain types of polyps
- A personal history of inflammatory bowel disease (ulcerative colitis or Crohn's disease)
- A family history of a hereditary syndrome such as familial adenomatous polyposis (FAP) or Lynch syndrome
- A history of radiation to the abdomen (belly) or pelvis to treat a prior cancer

It is easier to have other indications for high risk patients to alert/decide for early screening. That leaves the question to ask, how early should ones who are high risk to take a screening colonoscopy. Would it be great to have a study for any indications or similarity of younger patients who come into screening earlier, and concludes any indications for age in CLC screening. This section will address those concern and is the pioneering first step.

Within the 340 participants in this study, there are only 6 patients who are under 40 years of age. The rest of 334 participants are over 40 years of age. Obviously there are more older age patients in this study, however, if there is any trend in the classification and detection that can be helpful to do DNA profiling to further the study.

Figure 3.20 shows the overall result of the classification of polyps that broken down into size. The accuracy of the model for polyps in patients who are under 40 years old is 97.54%.

		Condition			
		Gold Standard:	Pathologist		
	Total Population TotPop=CP+CN =TestP+TestN 489	Condition Positive CP = TP + FN 270	Condition Negative CN = TN + FP 219	Prevalence Prev = CP/TotPop 0.552147239	
Test Outcome	Test Positive TestP = TP + FP 276	True Positive TP 267	False Positive Type 1 Error FP 9	Positive Predictive Value, Precision PPV = TP/TestP 0.967391304	False Discovery Rate FDR = FP/TestP 0.032608696
	Test Negative TestN = TN + FN 213	False Negative Type 2 Error FN 3	True Negative TN 210	False Omisson Rate FOR = FN/TestN 0.014084507	Negative Predictive Value NPV = TN / TestN 0.985915493
	Accuracy ACC = (TP+TN)/TotPop 0.97546012	Sensitivity True Positive Rate TPR = TP /CP 0.988888889	Fall-out False Positive Rate FPR = FP/CN 0.04109589	Positive Likelihood Ratio LR+ = TPR/FPR 24.06296296	Diagnostic Odds Ratio DOR = LR+/LR- 0.278824809
		Miss Rate False Negative Rate FNR = FN /CP 0.011111111	Specificity True Negative Rate TNR = TN/CN 0.95890411	Negative Likelihood Ratio LR- = TNR/FNR 86.30136986	

Figure 3.20: GI confusion matrix for polyps in patients who are over 40 years old

CHAPTER FOUR

Particle Identification in Radiation Therapy

Particle beams like protons and heavier ions offer improved dose distributions compared with photon (also called x-ray) beams. Proton and ion beams are commonly used worldwide for radiation therapy to treat cancer worldwide, offering advantageous dose distribution and increased relative biological effectiveness (RBE) compared to photons. The RBE is assumed to vary with linear energy transfer (LET) of particles, currently only the constant RBE is taken into account in treatment planning due to a lack of tools for its experimental validation.

Deep CNNs have come to dominate image recognition and identification tasks due to its ability to discover structures even in high dimensional data [33] and thus are the natural choice for particle identification on Timepix detectors. In particular, the VGG-16 network [34] was selected, since it addressed the problem of network depth in CNN by simultaneously increasing the number of convolutional layers and reducing their size to a 3×3 convolutional filter with stride 1. The VGG-16 network thus reduced parameters and still effectively performed larger convolution filters resulting in an elegant and very accurate convolutional network for recognition, classification, and localization [35].

4.1 Data Collection

Extensive measurements were performed in a gantry treatment room of Krakow proton therapy facility with pencil beams of various energies. A compact Timepix MiniPIX detector was protected by a waterproof cover and mounted inside a water phantom as shown in Figure 4.1 allowing accurate positioning with respect to the beam.

The first measurements were performed in air with beam nominal energies of 70 MeV, 100 MeV, 150 MeV and 200 MeV. Measurements were performed for detector set at various

angular positions for incident angle β equal to 30°, 45°, 60° and 75°. After that, the water phantom was filled and measurements were performed for proton beam of nominal energy of 150 MeV without the range shifter, as well as of energy 171.66 MeV with range shifter. Both combinations of the nominal energy and RS usage exhibit the same Bragg peak range in water of 156.6 mm.

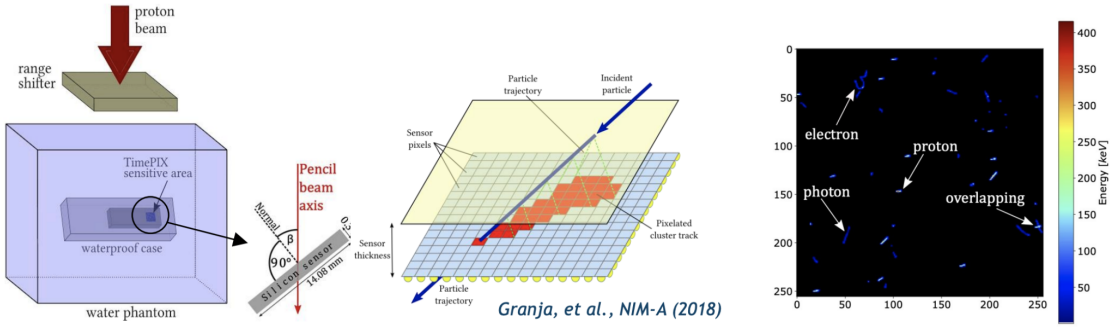

Figure 4.1: Experimental setup

4.1.1 Timepix Detector

Timepix is extensively used in radiation research[36]. Timepix detectors consist of a semiconductor layer (sensitive volume of the detector) bumb-bonded to the highly-integrated ASIC readout chip. The detectors provide a high-granularity array pf closely packed pixels with independent signal readout electronics per pixel. The hybrid architecture allows to apply various materials and thicknesses of the semiconductor. Thickness of the semiconductor layer is usually within the range of 100 to 2000 μm. The Timepix chip is an array of 256 $\times$ 256 pixels with individual readout electronics per pixel[37]. Each pixel has size of 55 $\times$ 55 μm, which gives a sensor sensitive area of 1.96 cm^2. Clusters are produced by different particles in mixed radiation field of proton pencil beam in water. Low-LET, narrow, curly tracks are typical for electrons, high-LET, wide, straight tracks for energetic heavy charged particles such as protons, while low-LET, straight tracks are characteristic for photons. In the right side of Figure 4.1 an example of overlapping clusters is shown. Timepix is calibrated so that the signal is proportional to the energy deposited in each pixel

and the response of all pixels is unified. Signal acquisition time in the entire matrix of pixels has to be adjusted by the operator of the detector for each measurement (usually from 1 to 200 ms). The readout of the signal is performed separately in each pixel and the deadtime depends on the generation of Timepix.

In this project, a Timepix detector was used with a 300 μm silicon sensor operated and readout with integrated electronics MiniPIX TPX [38]. The detector has small and compact dimensions of 77 mm × 21 mm × 10 mm and it has a weight of 25 g). This detector requires a single USB 2.0 connector for integrated power, control and readout. The detector operates at room temperature without need of active cooling and with a water-proof seal can be immersed in water. The Timepix chip in frame mode and pixel-level configuration in energy mode [37]. The detector was energy calibrated per pixel [39]. The camera is controlled and readout by the cross-platform software PIXET (Advacam, Prague) which provides online visualization of data acquisition and high data rate (up to 45 frames per second and readout deadtime 22 ms). The signal is generated when the ionizing particle traverses the sensitive volume of the detector producing an electric charge, which is collected in nearby pixels readout. The signal collected in the adjacent pixels forms a cluster. The morphology of the produced cluster of pixels (the trace of the particle) is characteristic for the type, incident angle and the energy of the ionizing particle that induced the signal. Knowing the energy deposition in the cluster and analyzing its morphology it is possible to recognize the type of the particle and to compute its LET.

4.1.2 LETs Computation

The LET of protons is computed by traversing the Timepix sensor the same way for experimental and simulation data as the ratio of energy imparted by the particle in the sensor, ϵ, and the track length, l, of the particle in the sensor:

$$\mathrm{LET} = \frac{\epsilon}{\mathrm{l}} \quad . \tag{4.1}$$

The track length was computed assuming that the particle does not stop in the sensor volume based on the cluster length in the sensor plane and sensor thickness as follows:

$$l = \sqrt{{l_{2D}}^2 + d^2} \quad , \tag{4.2}$$

where l_{2D} is the length of the cluster derived from the cluster morphology and d is the thickness of the silicon sensor .

Most often conversion of LET in silicon to water or tissue is carried out using a constant conversion factor [40, 41, 42, 43]. A fitted function proposed by [44] is applied to cover a wide range of energies and is considered to be a more accurate approach.

Dose-averaged LET, LET_D, in radiobiological models applies only to primary and secondary protons, therefore, protons were separated from other secondary particles. This allowed to compute LET_D only for protons using the following formula [45]:

$$LET_D = \frac{\sum_i \left(\frac{\varepsilon_i}{l_i}\right) \varepsilon_i}{\sum_i \varepsilon_i} \quad , \tag{4.3}$$

where i is the particle index.

LET measurement uncertainty was propagated taking into account the energy deposition, sensor thickness and particle incident angle uncertainties. Energy deposition uncertainty was computed based on the fit to experimental data. Sensor thickness uncertainty estimated by the manufacturer is $\pm\, 10\,\mu$m and the angle uncertainty estimated based on MC simulations is 2°. The uncertainty of LET conversion from silicon to water is not provided [44].

4.2 Methodology

Particles were identified using a deep convolutional neural network. The network was trained using transfer learning, with initial training from the pre-trained Keras VGG16 model [34], and subsequent training using a homogeneous data set of proton, electron, and photon clusters.

The labeled data set was provided by Advacam and composed of 2,899,816 clusters for protons, 853,717 clusters for electrons, and 3,143,149 clusters for photons (X rays, gamma rays). Data were collected in well-defined radiation fields provided by radionuclide sources and particle accelerator beams of light and heavy charged particles: 4-25 MeV electrons from the MT-25 Microtron accelerator at the Nuclear Physics Institute (NPI), Czech Academy of Sciences (CAS), Rez near Prague, 8-35 MeV protons from the U-120M cyclotron of the NPI-CAS Rez, 70-200 MeV protons from the CCB cyclotron at IFJ PAN Krakow. X rays and low-energy gamma rays were measured using ^{241}Am source at Advacam. Gamma rays were measured also using ^{60}Co and ^{137}Cs radionuclide irradiators at the Czech Metrology Institute (CMI) Prague. Description of these calibration measurements and of the data acquired with a similar Timepix detector with a 300 μm silicon sensor are given elsewhere [46, 47, 48, 49].

Data was appropriately filtered to remove overlapping clusters, clusters produced by secondary particles or at the sensor edge, as well as detector artifacts. Pre-processing of calibration data follows five steps. At first, per pixel energy calibration happened to make the conversion of time-over-threshold into energy in keV. Clusterisation is grouping of pixels to sets/clusters belonging to individual particle tracks based on coordinate/spatial vicinity. Cluster analysis is to calculate the morphological and spectral features of clusters. Pile-up filtering applied to filter the pile-up based on morphological features and on additional per pixel energy conditions. Lastly, the additional filtering happened to remove the secondary particles, clusters at sensor edge, and detector artifacts (noisy pixels, not fully discharged pixels etc.) with conditions on cluster features.

The pre-trained VGG16 model was chosen because it is a deep network that achieved good results in both image classification and localization [35] - promising for blob like structure [50]. Since there were more than 3 times as many protons as electrons in the data set, a generative adversarial network (GAN) was used to augment the electron clusters [51, 52]. To avoid mode collapse in GAN training, instance noise was used [53].

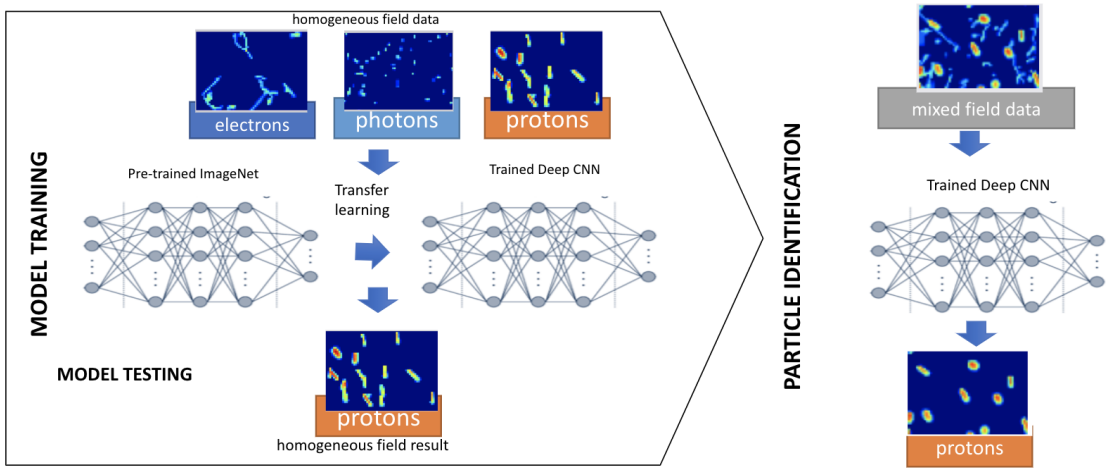

Figure 4.2: Overview of the AI PID flow

There are many proposed GAN framework in medical image research such as Deep Convolutional Generative Adversarial Networks (DCGAN), Laplacian Generative Adversarial Networks (LAPGAN), pix2pix, Cycle-GAN, UNIT, CatGAN, BigGAN, InfoGAN, VAEGAN, StyleGAN and more. In this work, a DCGAN [54] framework was adapted with electron cluster data set to address the class imbalance problem in the training data set as DCGAN was proven to be effective in generating total electron content maps [55] and it is more stable to train[54] than the original GAN.

The DCGAN network parameters are optimized by the equation,

$$\min_{G}\max_{D} V(G,D) = \mathop{\mathbb{E}}_{§\sim\rho_{data}(§)}[log(D(x))] + \mathop{\mathbb{E}}_{‡\sim\rho_{z}(‡)}[log(1-D(G(x)))] \tag{4.4}$$

in which D and G are the discriminator and generator functions respectively. x is the sample from the data distribution, P, z is a random variable following the distribution P_Z, and $\mathbb{E}$ is the expectation operator.

The CNN was trained for 100 epochs toward convergence. The data of electron clusters created by DCGAN was used for training only. The data set was randomly divided into training and testing using 10-fold cross validation [56]. Accuracy and uncertainty were assessed using confusion matrices [57]. After the trained model was established, the

mixed radiation field data set was used as the input in order to identify protons from other particles.

4.3 Results

4.3.1 Homogeneous Data Set

The result for homogeneous data set is presented in this section. The metrics to determine the quality of the model is the same as in Chapter 4. Accuracy, precision, sensitivity, and specificity were used to evaluate performance of the model [57]. The accuracy is the ratio of the correct predictions to the particles in the homogeneous data set [57]. The accuracy achieved for proton recognition is 95.36% as shown in Figure 4.3. This is a good for a model because it gives high confidence. The resulting neural networks have sensitivity of 92.97% and specificity of 97.10%. The NPV in this result is at 95.01%.

For the performance of the model in mixed radiation field data set, LET spectra were computed and compared with MC simulation to show the accuracy as the mixed field data set was not labeled, hence the ground truth was an unknown.

		Condition			
		Gold Standard: Labeled Particles			
	Total Population TotPop=CP+CN =TestP+TestN 6896682	Condition Positive CP = TP + FN 2899816	Condition Negative CN = TN + FP 3996866	Prevalence Prev = CP/TotPop 0.420465377	
Test Outcome	Test Positive TestP = TP + FP 2812148	True Positive TP 2,696,116	False Positive Type 1 Error FP 116,032	Positive Predictive Value, Precision PPV = TP/TestP 0.958739014	False Discovery Rate FDR = FP/TestP 0.041260986
	Test Negative TestN = TN + FN 4084534	False Negative Type 2 Error FN 203700	True Negative TN 3,880,834	False Omisson Rate FOR = FN/TestN 0.04987105	Negative Predictive Value NPV = TN / TestN 0.95012895
	Accuracy ACC = (TP+TN)/TotPop 0.95363974	Sensitivity True Positive Rate TPR = TP /CP 0.929754164	Fall-out False Positive Rate FPR = FP/CN 0.029030746	Positive Likelihood Ratio LR+ = TPR/FPR 32.02653411	Diagnostic Odds Ratio DOR = LR+/LR- 2.316994757
		Miss Rate False Negative Rate FNR = FN /CP 0.070245836	Specificity True Negative Rate TNR = TN/CN 0.970969254	Negative Likelihood Ratio LR- = TNR/FNR 13.82244565	

Figure 4.3: The confusion matrix for particle identification for homogeneous data set

4.3.2 Mixed Radiation Field Data Set

LET spectra measured in air with silicon sensor for therapeutic proton pencil beams for different incident angles of protons, namely 30°, 45°, 60° and 75° were compared. A good agreement of the LET spectra for various angles for all the considered beam energies was shown. For the lowest measured beam energy of 70 MeV an 8% shift of the peak (the most probable value) relative to the spectrum measured in 45° (considered the best) is observed for the largest angle of 75°. A percent of protons hitting the sensor at the angle of 75° or larger for all mixed field data measured by us is lower than 2%.

LET spectra verification in water was measured and compared for the proton beam nominal energies of 150 MeV and 171.66 MeV with range shifter at the depth of 149 mm in the beam axis. As for 171.66 MeV proton beam with application of range shifter the beam range should be the same as for 150 MeV proton beam, the LET spectra in the considered

points should be matching. Measured LET spectra produced by protons overlap within uncertainty ranges.

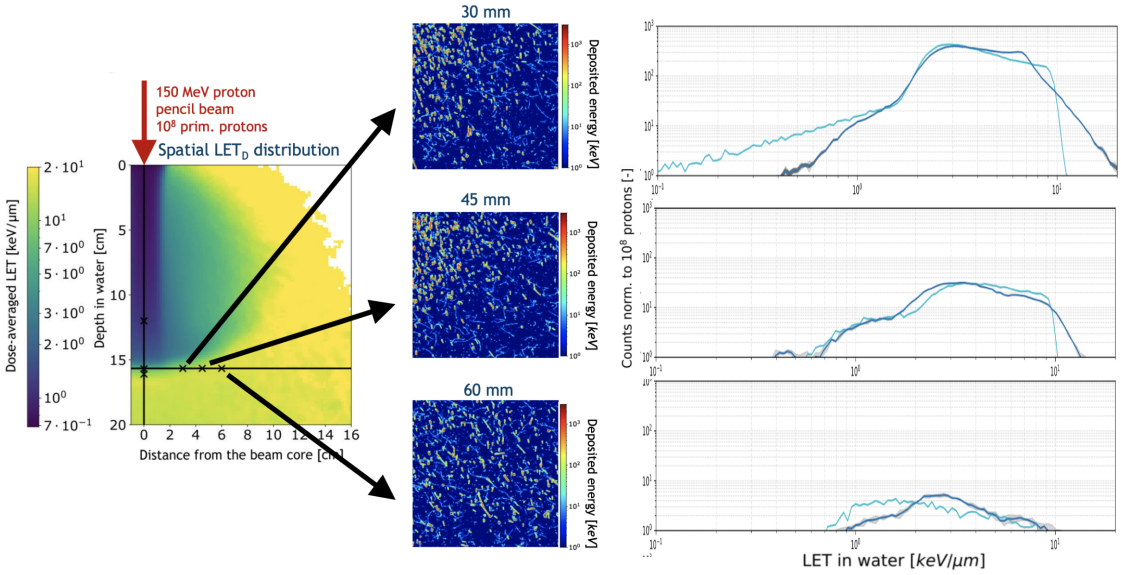

Figure 4.4: LET spectra measured (light line) and simulated (dark line) for 150 MeV proton pencil beam in water along lateral beam profiles at the depth of BP which is 156.6 mm. Measurements and simulations were performed at distances from the beam axis of at 30 mm, 45 mm and 60 mm for the profile at a depth of 156.6 mm (BP).

LET spectra along beam profiles in water are shown in figure 4.5, figure 4.6, figure 4.8, and figure 4.7. The relative number of protons decreases with distance from the beam axis as the height of the graph get smaller. Positions of the peak obtained in measurements and in simulations are overall in good agreement for small distances, although a shift occurs in case of farther points. The relative number of protons does not change significantly with depth, which reflects in the change in height of the graph, but is several orders of magnitude smaller at a distance of 37 mm with respect to measurements in the beam axis. Positions of the peak obtained in measurements and in simulations are overall in good agreement, but discrepancies can be observed in the plateau region.

Figure 4.5 presents measured and simulated LET spectra for points along lateral profiles of 150 MeV proton beam in water at the depths of 78.3 mm and distances from the beam axis of 30 mm, 60 mm and 90 mm for the profile. For measurements performed at a

depth of 78.3 mm LET_D values are 2.5 keV/μm, 2.8 keV/μm and 2.8 keV/μm at distances from the beam axis of 30 mm, 60 mm and 90 mm respectively. The corresponding MC simulations LET_D values are 2.3 keV/μm, 2.7 keV/μm and 3.1 keV/μm. The result shows relatively good agreement between the measurements and MC simulation since the largest relative difference between measurements and MC simulations of 12%.

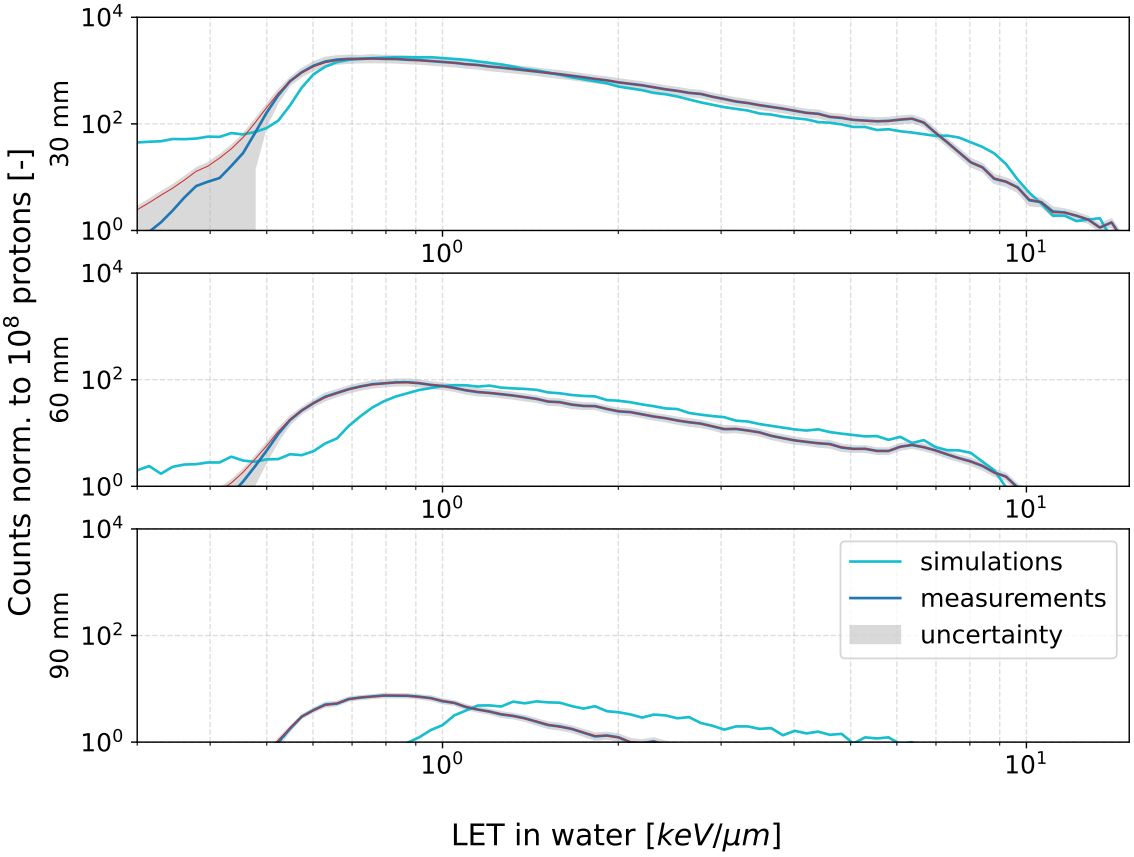

Figure 4.5: LET spectra measured (light line) and simulated (dark line) for 150 MeV proton pencil beam in water along lateral beam profiles at the depth of BP which is 156.6 mm. Measurements and simulations were performed at distances from the beam axis of at 30 mm, 45 mm and 60 mm for the profile at a depth of 156.6 mm (BP).

Figure 4.6 presents measured and simulated LET spectra for points along lateral profiles of 150 MeV proton beam in water at the depths of 156.6 mm which is BP depth, and 30 mm, 45 mm and 60 mm for the profile. For measurements performed at a depth of 156.6 mm LET_D values are 9.1 keV/μm, 5.5 keV/μm and 6.0 keV/μm at distances from the

beam axis of 30 mm, 45 mm and 60 mm respectively. The corresponding MC simulations LET_D values are 5.0 keV/μm, 5.3 keV/μm and 3.9 keV/μm. The large discrepancies at a level of 50-80% can be a result of long tails in the high LET range for measurements which are not present in MC simulation results.

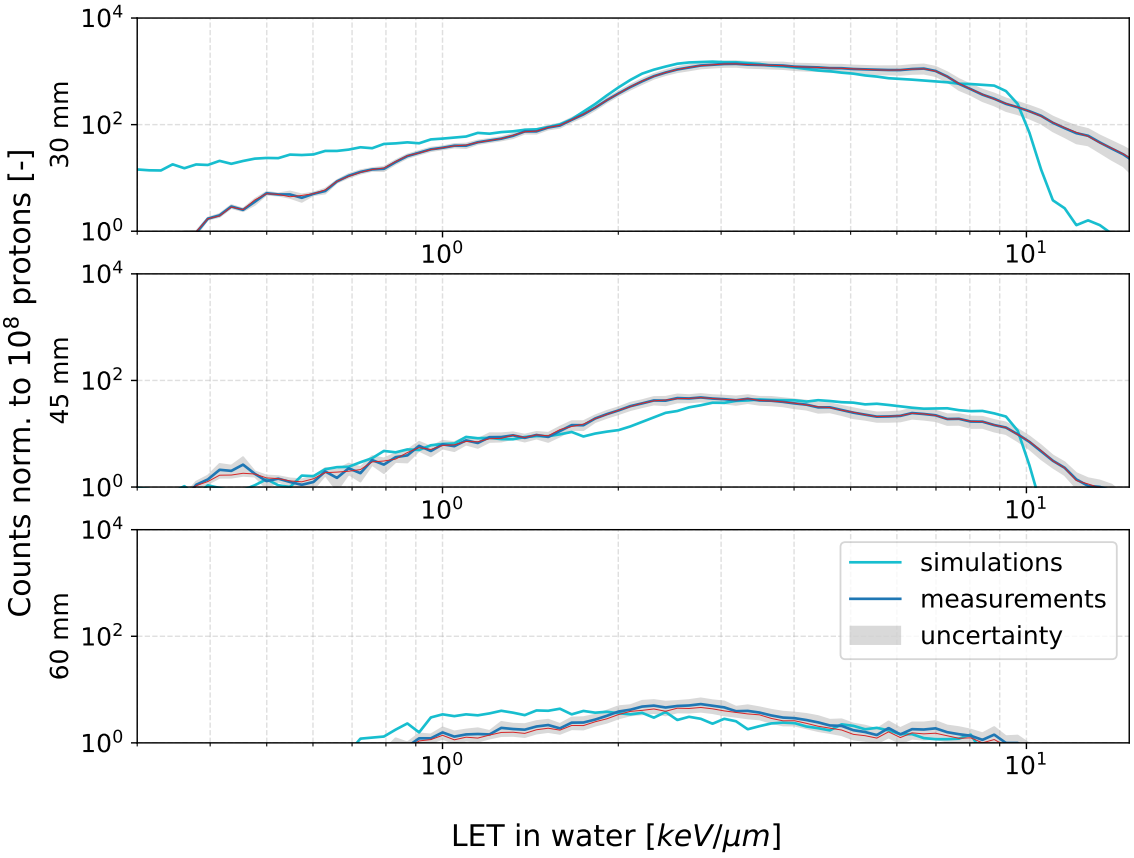

Figure 4.6: LET spectra measured (light line) and simulated (dark line) for 150 MeV proton pencil beam in water along lateral beam profiles at the depth of 78.3 mm. Measurements and simulations were performed at distances from the beam axis of 30 mm, 60 mm and 90 mm for the profile at a depth of 78.3 mm.

Figure 4.7 presents measured and simulated LET spectra for points along longitudinal profiles of 150 MeV proton beam in water in the beam axis. The results are presented for measurements and simulations at depths in water of 30 mm, 120 mm and 156.6 mm and 161 mm for this profile. For measurements performed in the beam axis, LET_D values are 1.1 keV/μm, 1.3 keV/μm, 4.0 keV/μm and 5.1 keV/μm at depths in water of 30 mm,

120 mm, 156.6 mm and 161 mm respectively. The corresponding MC simulations LET_D values are 1.2 keV/μm, 1.2 keV/μm, 4.3 keV/μm and 5.2 keV/μm. The largest relative difference for these measurements is at a level of 6%. The result shows the strong agreement between the LET spectra from particle identification model and the MC simulation.

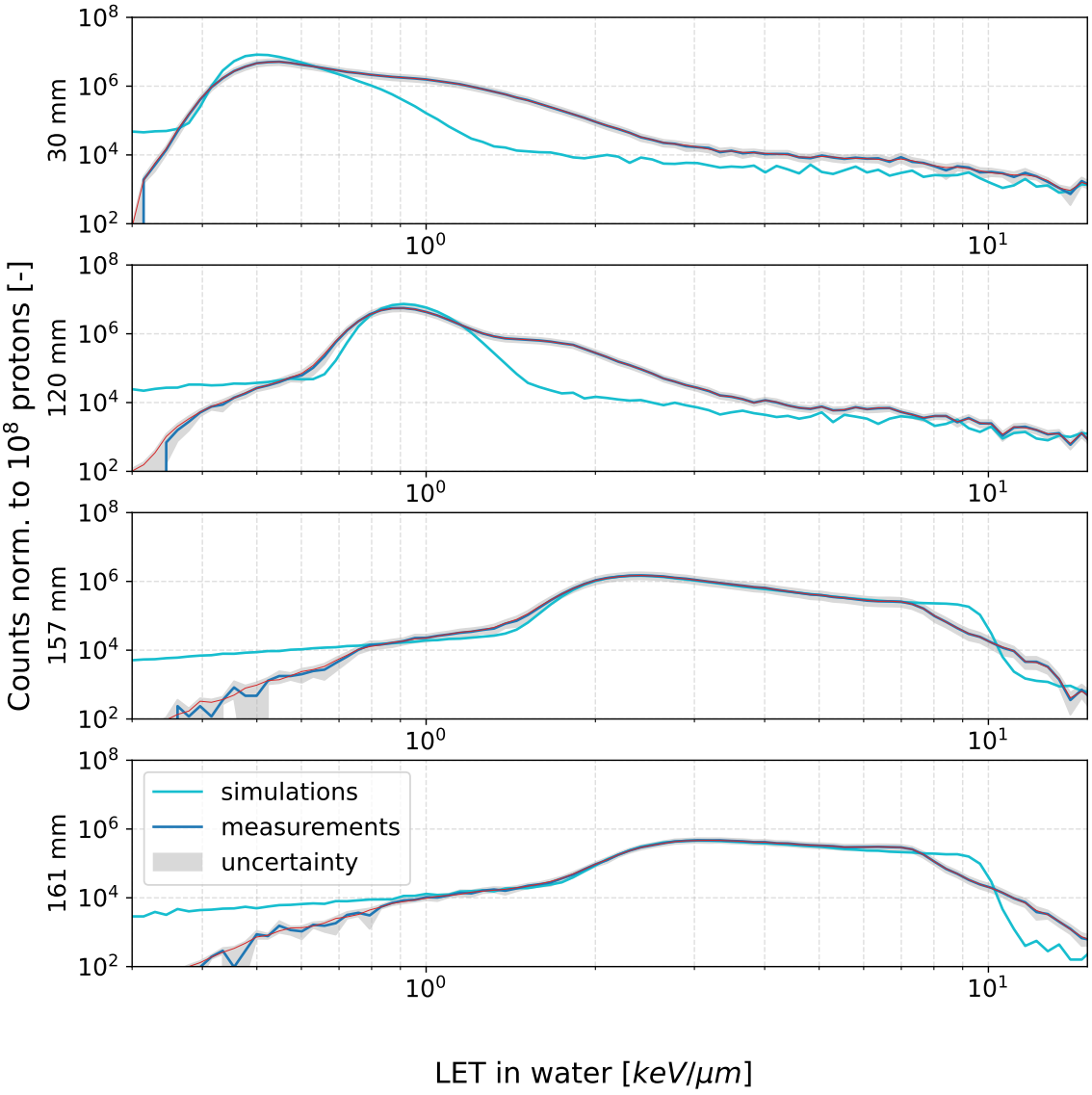

Figure 4.7: LET spectra measured (light line) and simulated (dark line) for 150 MeV proton pencil beam in water along longitudinal beam profiles in the beam axis . Measurements were performed at the depths of 30 mm, 120 mm, 156.6 mm (BP) and 161 mm.

Figure 4.8 presents measured and simulated LET spectra for points along longitudinal profiles of 150 MeV proton beam at a distance of 37 mm from the beam axis. The results are presented for measurements and simulations at depths in water of 30 mm, 120 mm and 156.6 mm and 161 mm for this profile. The measured LET_D values are 3.0 keV/μm, 2.7 keV/μm, 5.3 keV/μm and 6.5 keV/μm at depths in water of 30 mm, 120 mm, 156.6 mm and 161 mm respectively. The corresponding MC simulations LET_D values are 2.5 keV/μm, 2.4 keV/μm, 5.3 keV/μm and 5.5 keV/μm. This result can be looked at as the three part, the smallest relative difference is 0% at the depth of 156.6 mm. The largest relative difference for LET_D measurements is 20% at the depth of 30 mm.

The grey band in each graph is the uncertainty of the model. The uncertainty is the variance of the predictive probability[58]. The uncertainty is related to the identification of the particle. The uncertainty is not just the value of the strongest weight, but it is related to the variance of the predicted outcomes. To determine the variance of the predicted outcomes, the variance for the second to last layer was calculated and converted into percentage.

For the majority of the proton LET spectra in water, a lower number of protons was measured than was predicted by MC simulations at the low-end of the spectra. This is likely the error in particle recognition model, as there is a limited amount of proton data in this low energy range, and proton morphology in low range energy can look similar to electron in some cases. This could be improved by providing more training data specially for electrons and low-LET protons.

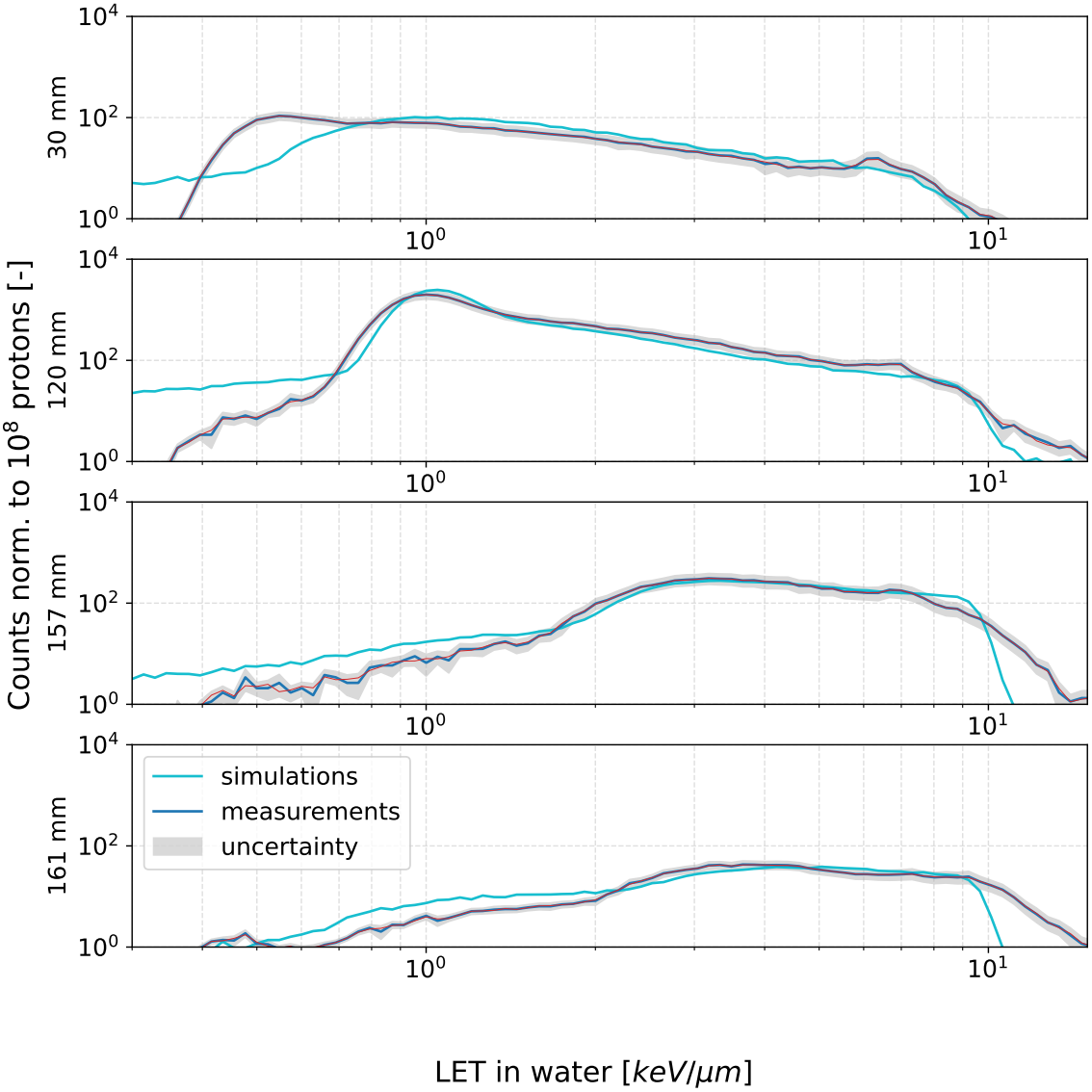

Figure 4.8: LET spectra measured (light line) and simulated (dark line) for 150 MeV proton pencil beam in water along longitudinal beam profiles out of the beam at a distance of 37 mm. Measurements were performed at the depths of 30 mm, 120 mm, 156.6 mm (BP) and 161 mm.

CHAPTER FIVE

Nitinol Wire

This chapter is based upon works published in the Society of Photo-Optical Instrumentation Engineers (SPIE) as part of L. J. Olafsen, B. Jones, L. Sparks, H. H. Nguyen, A. Tanner, K. E. Schubert, J. S. Olafsen, S. Dayawansa, E. Fonkem, J. H. Huang, "Current-controlled Nitinol Wire for Improved Arterial Navigation," SPIE Proceedings Vol. 10868: Advanced Biomedical and Clinical Diagnostic and Surgical Guidance Systems XVII (2019) and in IEEE journal, part of D. R. DeVries, L. J. Olafsen, J. S. Olafsen, H. H. Nguyen, K. E. Schubert, S. Dayawansa, and J. H. Huang, "Ultrasound Localization of Nitinol Wire of Sub-Wavelength Dimension," IEEE Open Journal of Engineering in Medicine and Biology 3, 18–24 (2022).

Nitinol is a shape memory alloy of nickel and titanium, with a long record of bio compatibility, particularly when an oxide and/or another passivating layer is applied [59]. Nitinol wires have been used widely as stents because the alloy can be programmed to expand at human body temperatures and solidly fix to blood vessel walls [59, 60].Ultrasound is a safe (non-ionizing), low-cost, portable, and widely available imaging modality that uses high frequency sound waves typically in the 1–10 MHz range. Lower frequencies (1–3 MHz) are used for deeper structures such as the liver or heart, and higher frequencies (5–10 MHz) are used for imaging near the skin [61]. To enhance vascular navigation using surgical guide wires and reduce the use of ionizing radiation, a method for ultrasonic localization of wires with diameters less than the wavelength of ultrasound in the phantom custom made with gelatin was studied. The data taken from that was used to train and test two CNNs to localize the wire.

5.1 Current Control of Nitinol Wire

This section is based on the first conference paper that was published in 2018. Nitinol wire is used with a programmed heat activation above body temperature, and thus guidance can be externally controlled using resistive heating. We present results of current-controlled steering of nitinol wire, including the programming, control, and material response to varying current levels and pulse durations [62]. Nitinol wires of 100 μm and 150 μm in diameters were activated then the angle test were performed to study the wire bending angle respect to varying currents in air and liquid.

To train the wire, the following steps were performed:

(1) Straighten the wire and place it on the hotplate at around 1000°F. Leave the wire for a few minutes.

(2) Using the soldering iron (heated to 850°F) pull on the middle of the wire so that it bends around the soldering iron tip

(3) Hold the soldering iron on the wire for a few minutes then remove the soldering iron

(4) Place the rounded end of the wire on the hot plate and place the tweezers about 1 inch from the point

(5) Bend the wire over the tweezers and hold that position for a few minutes

(6) Remove the wire from the hotplate and quench

The air testing was based on the training method above with slightly modification in the setup that the wire and the protractor were tapped onto flat surface to minimize movement and change in the experiment. In a shallow Pyrex dish, deionization (DI) water was filled to a depth of 5mm. The activated nitinol wire then was placed in the water near the surface. The wire was bend by applying a current for 45 seconds. The resulting angle was measured by a fixed protractor. The wire was allowed to return to the neutral position.

The process then was performed twice more. A setup with a fixed protractor on a flat surface was used to improve the consistency of a measuring procedure.

The water testing showed that it took longer with a higher current level to bend in water. The wire is thin, and the surrounding water cools the wire; hence more current required to heat the wire up. Water caused the cooling effect. The initial test was done in the extreme case that the different between temp of water and temp of transition was 50°C while in a clinical setting the difference of temperature would be only 5°C, therefore less current is required. A human blood vessel is much smaller and contains much less liquid than the Pyrex bath. The most challenging task was keeping the wire from touching itself and creating a short circuit. The protractor was not fixed in the water test because the distortion in the Pyrex was hard to read. Due to the necessary change in measurement style, the water testing is prone to error more than the air testing.

The result of bend angle versus injection current in air shown in Figure 5.1 and Figure 5.2 for 100 μm diameter wire and 150 μm diameter wire respectively. The current range required to achieve complete bending to the programmed shape (right angle) range for 100 μm diameter wire is from 70 mA to 150 mA for 100 μm diameter wire. The current range required to achieve complete bending to the programmed shape (right angle) range is from 225 mA to 300 mA for 150 μm diameter wire. The thinner wire requires less current for activation (bending). The trends for both diameters were notable. The amount of current required to activate the wire was proportional to the cross-sectional area of the wire.

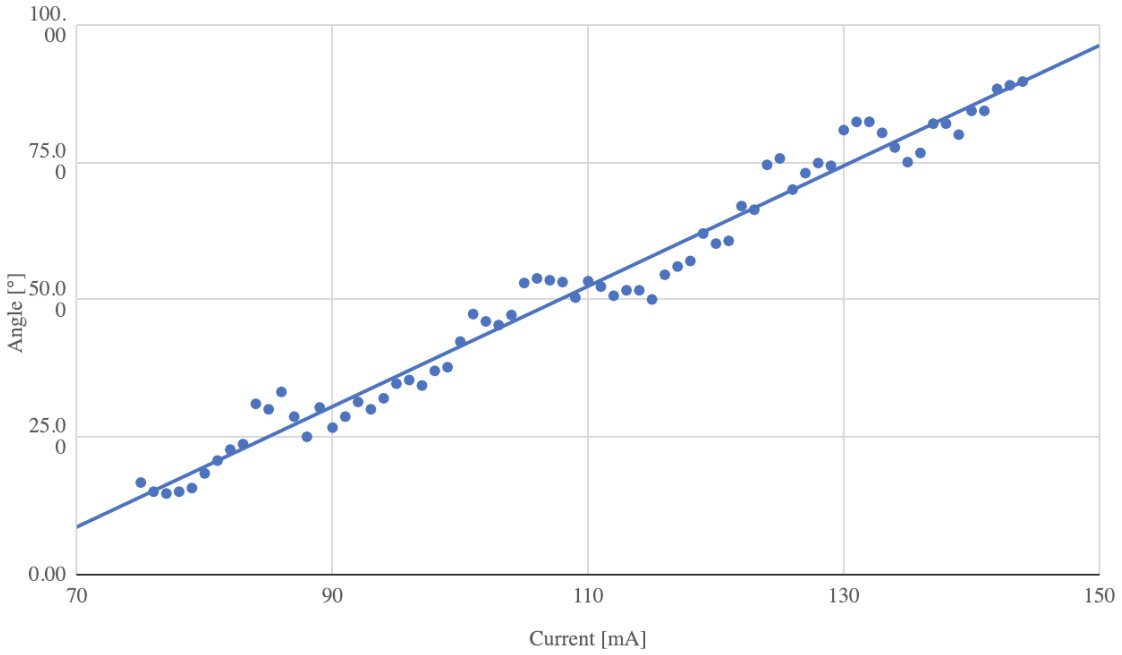

Figure 5.1: Angle test of 100 μm diameter wire in the air

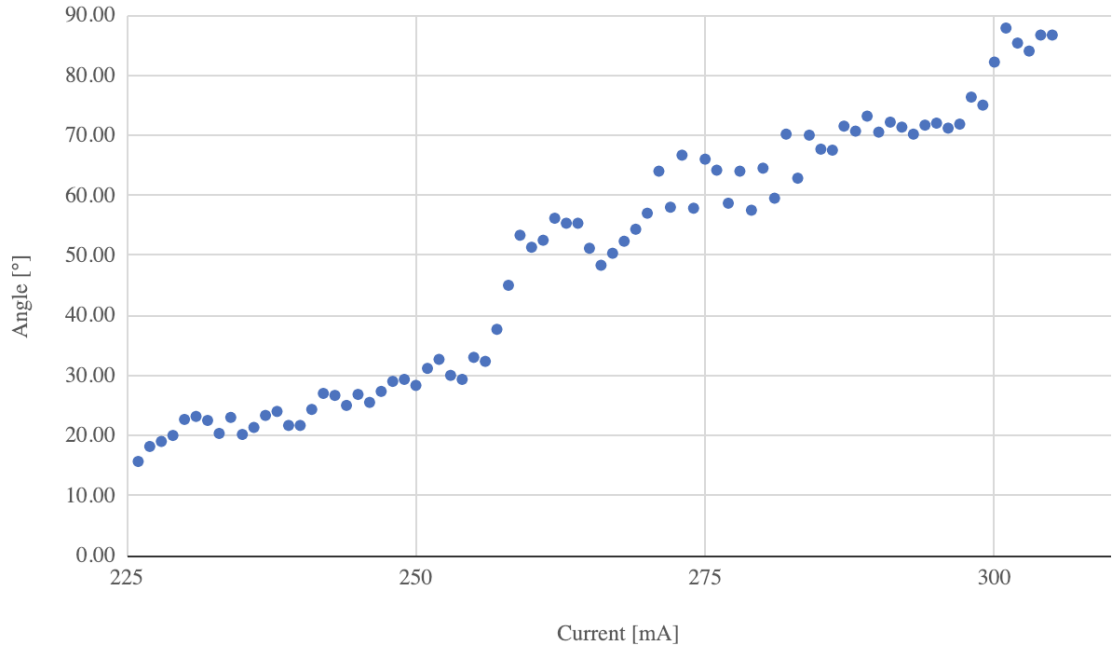

Figure 5.2: Angle test of 150 μm diameter wire in the air

The result of bend angle versus injection current in water shown in Figure 5.3 and Figure 5.4 for 100 μm diameter wire and 150 μm diameter wire respectively. The current range required to achieve complete bending to the programmed shape (right angle) range for 100 μm diameter wire is from 100 mA to 170 mA for 100 μm diameter wire. The current range required to achieve complete bending to the programmed shape (right angle) range is from 250 mA to 327 mA for 150 μm diameter wire. The consistency of the trends for both diameters was notable in water as well.

The preliminary test of nitinol wire and current is important to apply for future studies. The angle testing in air and water for 100 μm and 150 μm diameter wires shows that current levels are higher in water than in air to achieve the same bending angle. Lower current is desirable in arteries to minimize heating. Smaller wire diameters also are desirable as they support reduced minimum bend radii and thus allow for tighter control at turns and branches in the arteries. They also require lower currents and could provide access to smaller arteries.

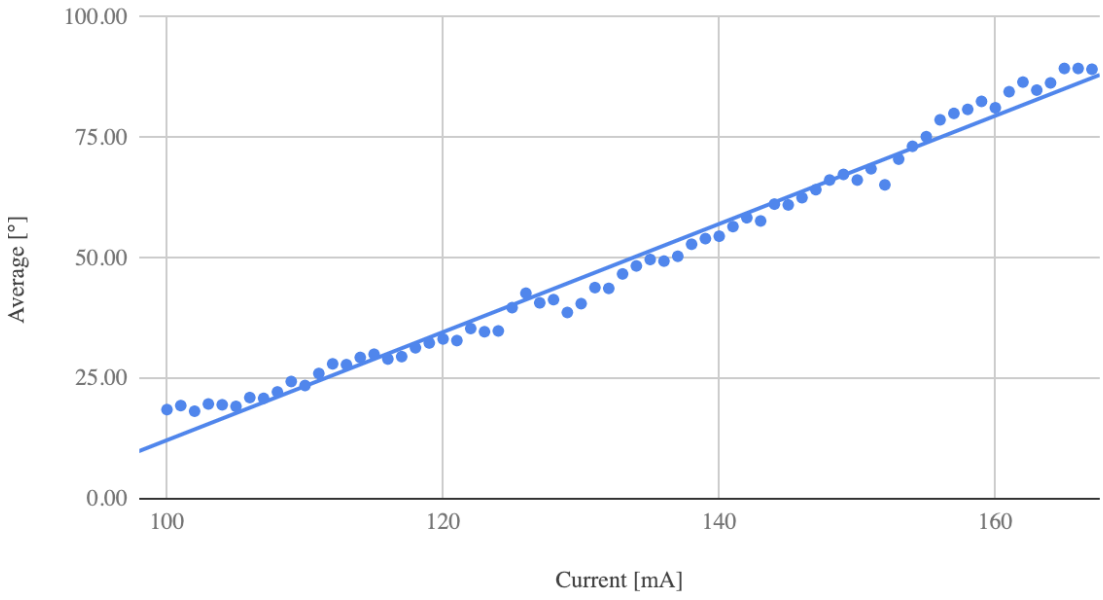

Figure 5.3: Angle test of 100 μm diameter wire in the water

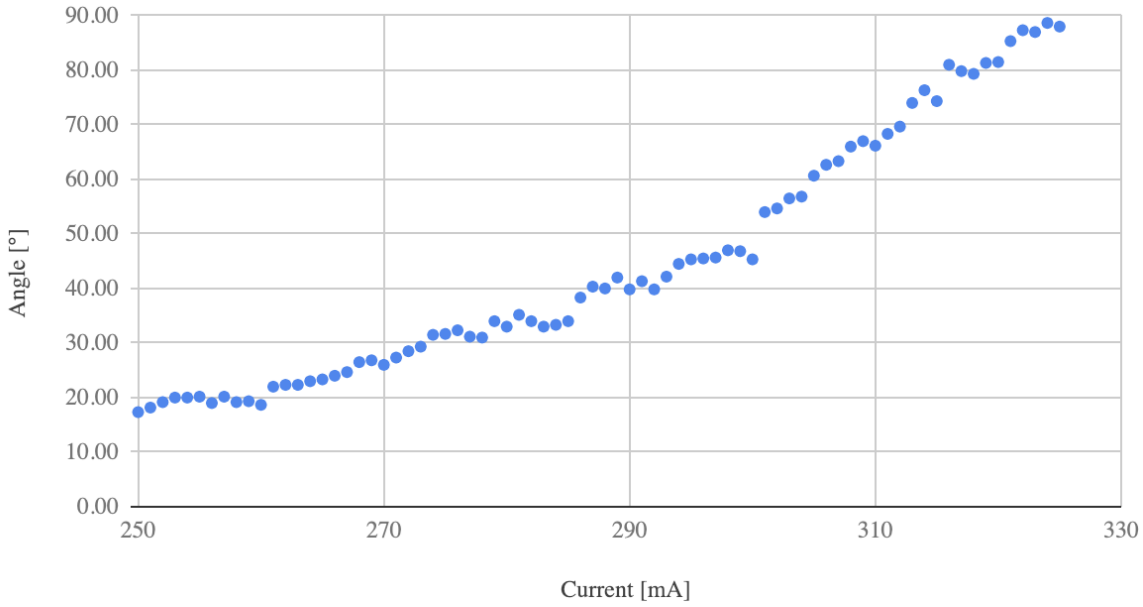

Figure 5.4: Angle test of 150 μm diameter wire in the water

5.2 Ultrasound Localization of Nitinol Wire

This section comes from the journal paper that was published in IEEE Open Journal of Engineering in Medicine and Biology 3 in 2021 and three patent submissions. In this project, gelatin phantoms were constructed in Dr. Linda Olafsen's lab by Daniella R. DeVries. Gelatin was chosen as material because it is excellent at mimicking tissue for ultrasound imaging [63, 64]. A rectangular slab phantom was created with different diameter tunnels through the center lengthwise to mimic vessels. The ultrasound probe was held midway along the phantom at position C0 as in Figure 5.5. The wire tip was advanced incrementally to each grid position with ultrasound images taken at position C0 to show progress of the wire through the phantom. Images were collected for four tunnel diameters (1/4 inch, 3/8 inch, 1/2 inch, 5/8 inch) and six wire diameters (50 μm, 75 μm, 100 μm, 125 μm, 150 μm, and 250 μm).

For the neural network analysis, a fine tuning VGG-16 model is executed to classify the ultrasound images with and without the nitinol wire in the feed through tubes of the gelatin phantom. We want to compare the original image versus

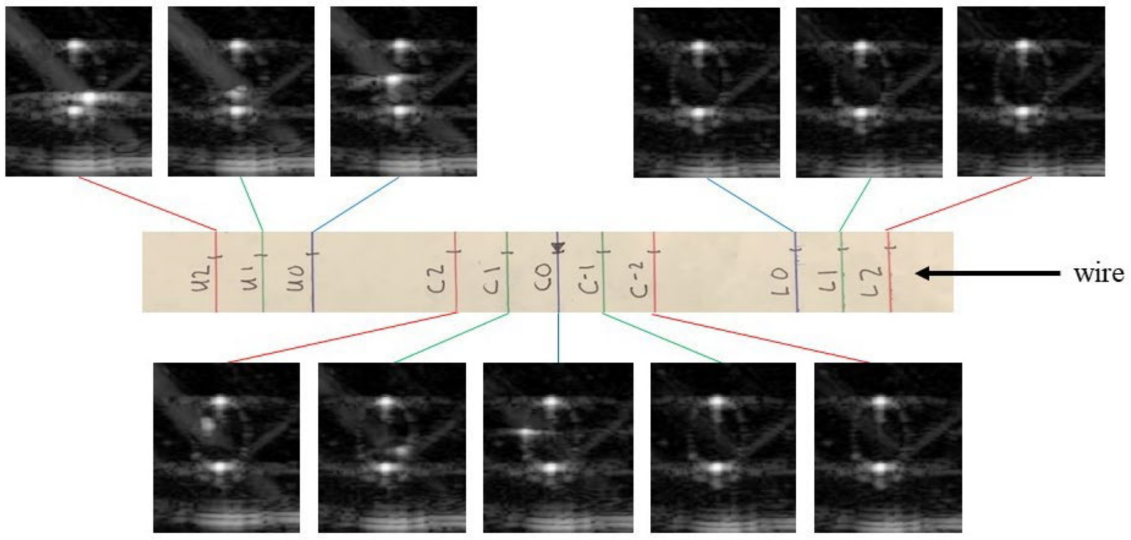

Figure 5.5: 75 μm wire in 1/4 inch diameter tunnel imaged at 7.5 MHz

Due to this, we constructed two data sets for the model. The first data set of 187 images (102 with wire and 85 without wire) is the original set of ultrasound images, including all six wire diameters and the three ultrasound frequencies. The second data set of 170 images (102 with wire and 68 without wire) is the set of images taking the difference between the regular image and the image at the L2 position.

Each data set is split into a training set and a validation set.

After importing the VGG-16 model, the three fully connected layers are discarded. Then the new specialized CNNs are trained using the gelatin phantom data sets. These specialized CNNs are small 3-layer with 3 by 3 kernels of numbers 32, 32, and 64. A regular sigmoid function is used to classify the images into two classes of with and without wire. The newly trained specialized CNN then is fed the features from the frozen VGG-16 model and the pre-trained weights are loaded into the model.

5.2.1 Results

The accuracy of the method is the ratio of the correct predictions to the total number of images in the data set. The overall accuracy achieved in each of the data sets is 95.19% and 96.47%. The resulting neural networks have sensitivity, specificity, positive predictive

value, and negative predictive value greater than 92%. The results from the difference image analysis are improved relative to the regular images, with respect to higher accuracy and narrower confidence intervals for accuracy, sensitivity, and positive predictive value. Table 5.1 shows the overall result of CNN training with corresponding 95% Confidence Intervals.

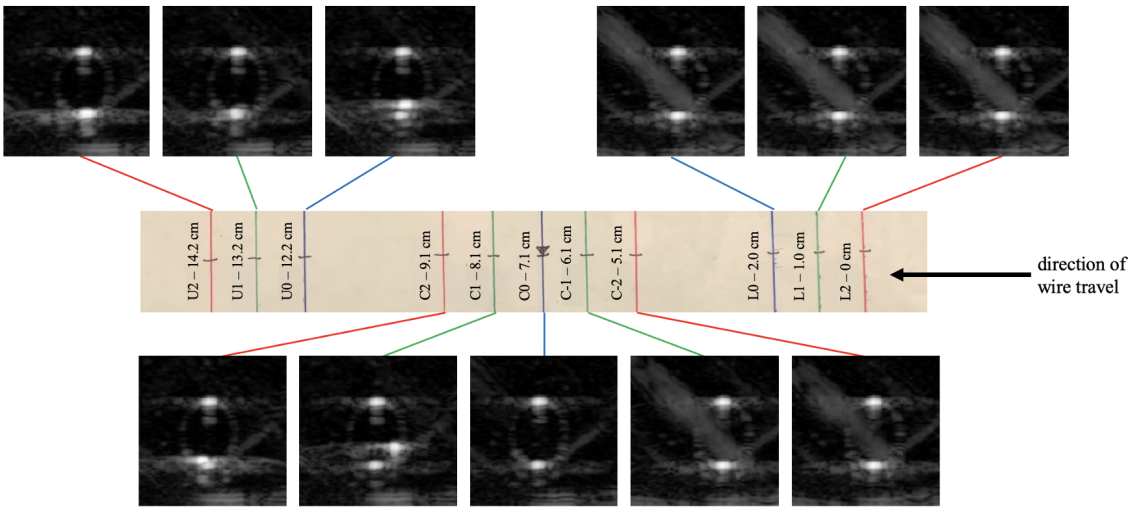

Figure 5.6: 50 μm wire in 1/4 inch diameter tunnel imaged at 7.5 MHz

Images are shown in Figure 5.6 at 7.5 MHz frequency as a 50 μm wire in 1/4 inch diameter tubing was advanced from position L2 to position U2. The ultrasound probe was fixed at position C0. Images were taken when the tip of the wire was at each of the labeled positions. The ultrasound localization technique to a wire of even smaller diameter.

A summary of results from the VGG-16 CNN model is showed as a confusion matrix in Figure 5.7 for both raw images and difference images relative to the image at position L2. In each matrix, the percentage is reported in each quadrant. True positive is shown in the upper left quadrant, with false positive in the upper right, false negative in the lower left, and true negative in the lower right. In seven cases, the CNN model applied to raw images yielded false negative results. The images corresponding to those cases are shown in Figure 5.8. Below each image are the imaging frequency, wire diameter, and wire

Table 5.1: Overall result of CNN on Ultrasound images

	REGULAR IMAGES	95% CI	DIFFERENCE IMAGES	95% CI
True Positive	95		97	
False Positive	2		1	
False Negative	7		5	
True Negative	83		67	
Accuracy	95.19%	0.9106-0.9778	96.47%	0.9248-0.9869
Sensitivity	93.14%	0.8823-0.9814	95.10%	0.90908-0.99288
Specificity	97.94%	0.9511-1.00766	98.53%	0.95668-1.0139
Positive Prediction Value	97.94%	0.9511-1.00766	98.98%	0.9699-1.00969
Negative Prediction Value	92.22%	0.86689-0.97755	93.06%	0.87184-0.98927

tip position for the raw image. There were five false negatives for the CNN analysis of difference images. The corresponding raw images are shown in all five cases in Figure 5.9. No false negatives occurred for 10 MHz images.

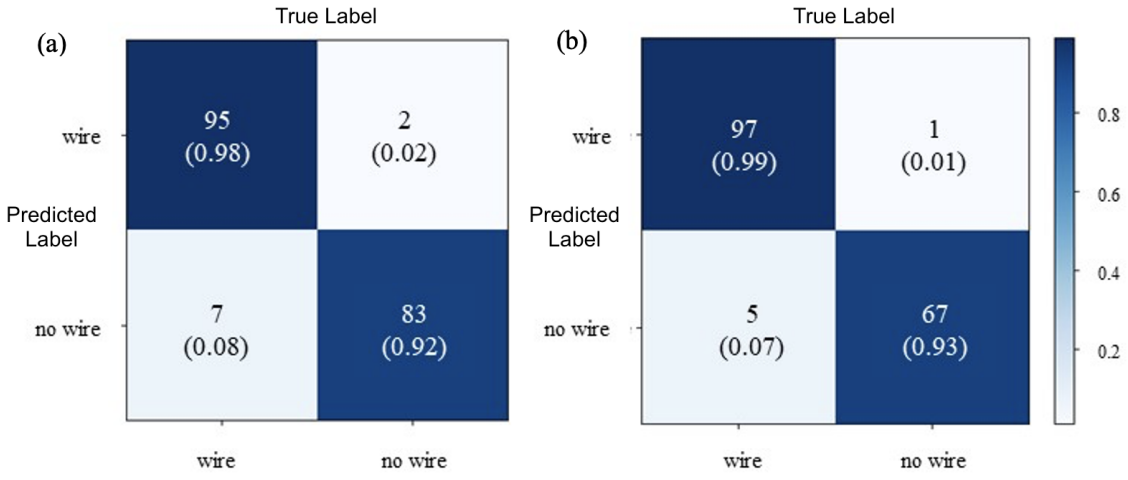

Figure 5.7: Confusion matrices obtained from raw images (a) and difference images (b) relative to the image at position L2

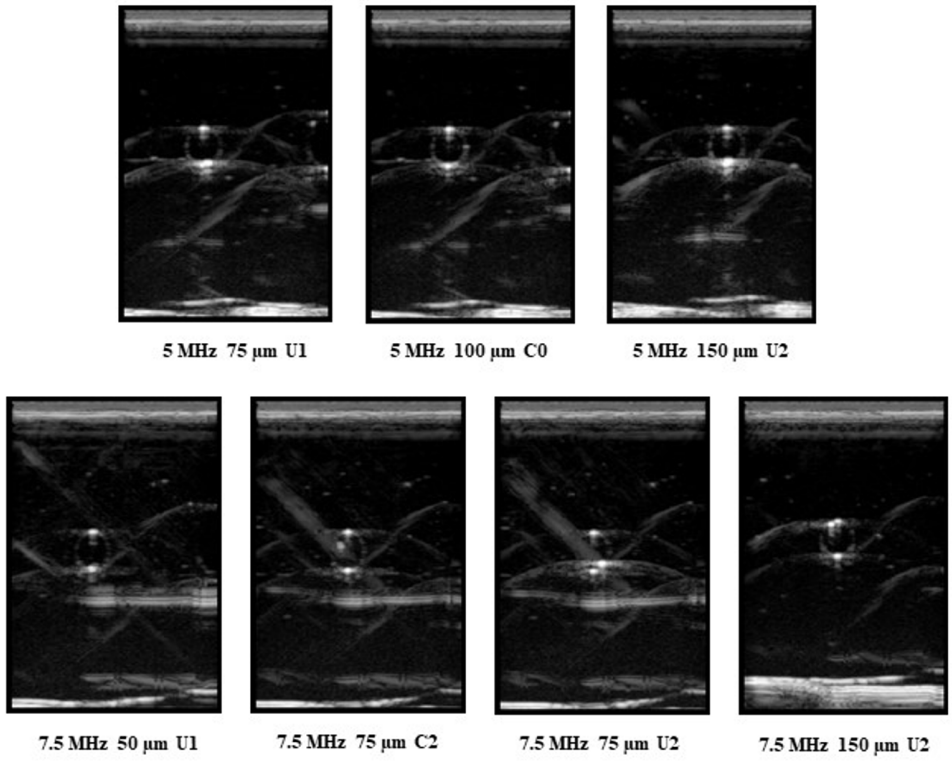

Figure 5.8: Ultrasound images for the false negative cases of the regular image CNN analysis.

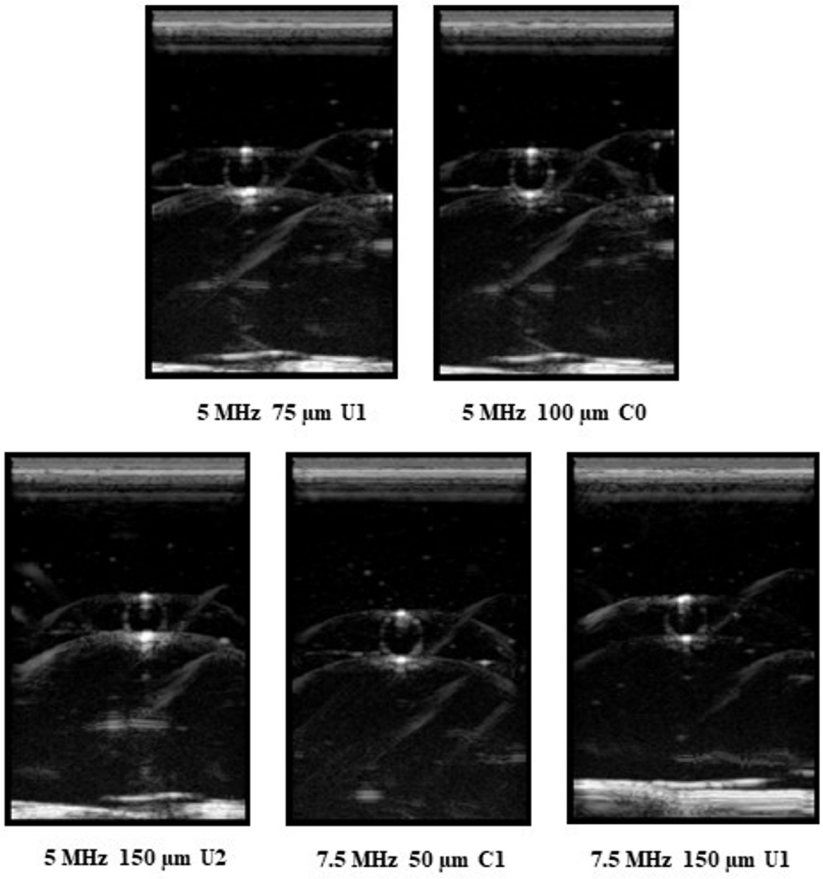

Figure 5.9: Ultrasound images for the false negative cases of the difference image CNN analysis.

CHAPTER SIX

Conclusion

In this dissertation, a deep CNN approach for biomedical application were presented and investigated in three distinct biomedical research areas. As a result, the publications are as following,

- 4 journals, one accepted, one submitted, two in preparation
- 3 patents pending
- 1 conference paper
- 3 clinical conference presentations
- 4 poster presentations

6.1 Automatic Classification of Colon Polyps

The first part of this dissertation, a polyp database in WLI and NBI was built by an IRB approved procedure. The images in that database were used to investigate the method of predicting polyp histology with CNNs. In the study, the automated analysis of multiple images of diminutive polyps obtained under both WLI and NBI has a high sensitivity in the diagnosis of adenomatous histology. The overall accuracy, sensitivity, and specificity of the model are greater than 97%. Additional it has 95.48% NPV, exceeding the 90% value set by ASGE. According to the clinical standard, this technology can and should be implement in clinical setting now. This technology can be applied to clinical clinics in remote areas all around the world.

Future work can add more polyps to the database, especially rectosigmoid and ascending polyps. A real time system can be built and tested in a clinical setting. Optimization of the performance for faster processing time should also be investigated.

6.2 Particle Identification in Radiation Therapy

In the second part of this work, a wide range of proton LET values in mixed radiation fields causes different complexity of DNA damage. Any LET-based variable RBE model requires accurate simulation and measurement methods for validation and quality assurance. An elegant and accessible approach to compare the agreement between measured and simulated LET spectra and dose-averaged LET of protons was introduced and investigated. The CNN for identification of particles, i.e., identify charged particles from other photons and electrons registered with Timepix detector was presented. This is an important application of AI for nuclear physics data analysis with direct implications for routine proton radiotherapy quality assurance and quality checks. CNN, particularly deep CNN, has come to dominate image recognition and identification tasks due to its ability to discover structures even in high dimensional data [33], and thus are the natural choice for particle identification on Timepix detectors.

The challenges overcame in this work include the class imbalance problem and the lack of feedback from the mixed radiation field identification in training the model. The class imbalance problem happens when the distribution of classes in the training data set is unequally distributed. The class imbalance problem negatively affects the CNN [65]. In this work, the number of electron clusters in the training data set is significantly less than the other two particles. This problem was addressed by using DCGAN to generate more electron clusters in training the CNN model.

However, another challenge of the training data set is that the beam nominal energy range is different for each type of particle in the training data set and the mixed radiation field. The training data was solely homogeneous; hence the mixed radiation field data was only used for testing. It should have been accounted for in the training part at the time of building the CNN model. The lack of feedback from the mixed radiation field identification also challenges the improvement of the method.

For future development, if more training data can be supplied to balance the set, it will help to address the imbalance problem. In addition, the non-visual data parameters should be included into the main network as the input to improve the identification. Since the current study is a supervised study, investigating a fusion of supervised and semi-unsupervised methods can improve and strengthen the current model's performance while still taking advantage of the learned features from the labeled data set. Heavy ions such as Helium and Carbon should be also included in the training process.

6.3 Nitinol Wire

In this dissertation's last part, a static estimation was investigated for the shaped alloy nitinol wire before moving to localize it in ultrasound. The result showed that a shaped alloy has the potential as a new active wire material in clinical medical applications. As the result of this work, an ultrasound instead of an invasive x-ray can be used to detect the location of the wire in a gelatin phantom as a proof of concept. This work has shown the CNN model can localize the wire. A CNN with transfer learning was applied to the original set of images and the difference set of images. It yielded a good result for the original image set and the difference image set showed even more potential with higher and more accurate results.

In the future study for the static estimation, the next step is to test how well the wire will bend in the tube with and without water. The tube is much like to human blood vessel. Human blood flows differently to water. It is also more viscous and slightly denser than water. Finding a solution with blood-like properties to test with the wire is also the next step. The next direction is to test the CNN model with another imaging modality, such as infrared can be examined with the same gelatin model.

APPENDIX

APPENDIX

Testing codes

1 Transfer Learning with VGG-16

```
import numpy as np
import keras
from keras import optimizers
from matplotlib import pyplot as plt
from numpy.random import seed
from tensorflow import set_random_seed
from keras.preprocessing.image import Imagedata_generator
from keras.models import Sequential
from keras.layers import Dropout, Flatten, Dense
from keras import applications
import os



from __futu__ import with_statement
im_width, im_height = 254, 254;

# train & validation dir
train_dir      = 'data_polyps/train'
validation_dir = 'data_polyps/validation'

modelFolder = 'VGG_models'
```

```
nb_train_samples      = 441
nb_validation_samples = 49

epochs     = 100
batch_size = 64

# training results using pre-trained VGG16
features_train_path = './VGG_models/
   transferVGG16_features_train.npy'

# validation results for polyps dataset using pre-trained
   VGG1
features_validation_path = './VGG_models/
   transferVGG16_features_validation.npy'

top_model_weights_path = './VGG_models/
   transferVGG16_fc_model.h5'

# calculate the outputs of VGG model for training and
   validation

def save_VGG_features(batch_size):
    # input: batch_size
    # output: number of the examples for the saved outputs (
       it could be different from the original dataset!)
    data_gen = Imagedata_generator(rescale=1./ 255)
```

```
# Build the VGG16 network without the top layer
model = applications.VGG16(include_top=False, weights='
    imagenet')

# generate training data from train folder
# generator = data_gen.flow_from_directory(
generator = data_gen.flow_from_directory(
     train_data_dir,
     target_size=(im_width, im_height),
     batch_size=batch_size,
     class_mode=None,
     shuffle=False)

# calculate the output of bottom VGG16 using training
    data
bottleneck_features_train = model.predict_generator(
         generator, nb_train_samples // batch_size)
# save the training outputs
np.save(VGG_features_train_path, VGG_features_train)

# generate validation data from train folder
generator = data_gen.flow_from_directory(
     validation_data_dir,
     target_size=(im_width, im_height),
     batch_size=batch_size,
     class_mode=None,
```

```
        shuffle=False)

    # calculate the output of VGG16 using validation data
    VGG_features_validation = model.predictionict_generator(
            generator, nb_validation_samples // batch_size)

    # save the validation outputs
    np.save(VGG_features_validation_path,
       VGG_features_validation)
    return (VGG_features_train.shape[0],
       VGG_features_validation.shape[0])

# train the top FC model from the pre-trained bottom VGG16
def train_top_model(epochs, batch_size, optimizer, drop_rate
   ):

    # load training outputs from the pre-trained bottom
       VGG16
    train_data   = np.load(VGG_features_train_path)
    train_labels = np.array([0] * (nb_train_samples // 2) +
       [1] * (nb_train_samples // 2))

    # load training outputs from the pre-trained bottom
       VGG16
    validation_data   = np.load(VGG_features_validation_path
       )
```

```
    validation_labels = np.array([0] * (
        nb_validation_samples // 2) + [1] * (
        nb_validation_samples // 2))

    # outputs from the bottom VGG16 + a FC layer
    model = Sequential()
    model.add(Flatten(input_shape=train_data.shape[1:]))
    model.add(Dense(256, activation='relu'))
    model.add(Dropout(drop_rate))
    model.add(Dense(1, activation='sigmoid'))

    model.fit(train_data, train_labels,
              epochs=epochs,
              batch_size=batch_size,
              validation_data=(validation_data,
                  validation_labels),
              verbose = 0,
              callbacks=callbacks)
    # save the final model
    model.save_weights(top_model_weights_path)
```

2 Preliminary Testing Code

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import glob
```

```
def read_data(particle_class, path):

    '''
    Classes:
    1 - proton
    2 - electron
    3 - photon
    '''

    elist_pathNames = glob.glob(path)

    col_names = ['CMLateral_Px', 'CMVertical_Px', 'Size_Px',
        'Energy_keV', 'Time_Min_ns', 'EnergyMax_keV', '
       PolarAng_Rad', 'L2D_Px', 'W2D_Px', 'WStd_Perp_Px', '
       WStd_Along_PX', 'Roundness', 'Linearity', '
       NBordIn_Div', 'Thickness', 'Thiness', 'CurlyThin']
    data = []

    for elist_pathName in elist_pathNames:
        dataSingle=pd.read_csv(elist_pathName, sep='\t',
           index_col=False, skiprows=1, names = col_names)
        data.append(dataSingle)

    data=pd.concat(data)
    data['Class'] = particle_class
```

```
    data.drop(['CMLateral_Px', 'CMVertical_Px', 'Time_Min_ns
        ', 'PolarAng_Rad'], axis=1, inplace=True)

    return data

data_proton = read_data(1, '/home/pid/Protons/*MeV_*/*deg/E
    *')
data_electron = read_data(2, '/home/pid/Electrons/*MeV/*deg/
    E*')
data_photon = read_data(3, '/home/pid/*/E*')

data_all = pd.concat([data_proton, data_electron,
    data_photon])

from sklearn.preprocessing import StandardScaler
scaler = StandardScaler()

data_features = data_all.drop('Class', axis=1) # features
scaler.fit(data_features)

data_scaled_feat = scaler.transform(data_features)
data_scaled_feat = pd.DataFrame(data_scaled_feat, columns =
    data_features.columns) # standardized features

from sklearn.model_selection import train_test_split
X = data_scaled_feat
y = data_all['Class']
```

```
X_train, X_test, y_train, y_test = train_test_split(X, y,
    test_size=0.3, random_state=42) # tran/test data

from sklearn.neural_network import MLPClassifier
mlp = MLPClassifier()
mlp.fit(X_train, y_train)

prediction = mlp.predict(X_test)

from sklearn.metrics import confusion_matrix,
    classification_report
print(confusion_matrix(y_test, prediction))
print(classification_report(y_test, prediction))

# mixed field
def read_data_mixed(path):

    col_names = ['CMLateral_Px', 'CMVertical_Px', 'Size_Px',
        'Energy_keV', 'Time_Min_ns', 'EnergyMax_keV', '
       PolarAng_Rad', 'L2D_Px', 'W2D_Px', 'WStd_Perp_Px', '
       WStd_Along_PX', 'Roundness', 'Linearity', '
       NBordIn_Div', 'Thickness', 'Thiness', 'CurlyThin']
    data=pd.read_csv(path, sep='\t', index_col=False,
       skiprows=1, names = col_names)
    data.drop(['CMLateral_Px', 'CMVertical_Px', 'Time_Min_ns
       ', 'PolarAng_Rad'], axis=1, inplace=True)
```

```
    return data

mixed = read_data_mixed('/home/shared/TimePix/meas/
   pixetAnalysedData/pid_calib_data_from_box/Mixed/150
   MeV_156.6mm_45mm/EList.txt')

mixed_scaled = scaler.transform(mixed)
mixed_scaled = pd.DataFrame(mixed_scaled, columns =
   data_features.columns)
mixed_pred = mlp.predict(mixed_scaled)

def getHist(data, bins=np.arange(0, 5 + 0.01, 0.01)):

    if not isinstance(data, pd.DataFrame):
        data = pd.DataFrame(data)

    hist = list(np.histogram(data.iloc[:, 0], bins=bins))
    hist[1] = hist[1][:-1] + np.diff(bins) / 2

    return [hist[1], hist[0]]

bins=np.logspace(0,5,200)
hist = getHist(mixed.Energy_keV, bins=bins)

plt.figure(figsize=(8, 5))
plt.plot(hist[0],hist[1], 'k-')
```

```
plt.xscale('log')

hist = getHist(mixed[mixed.Type==1].Energy_keV, bins=bins)
plt.plot(hist[0],hist[1], 'r-', label='Protons')

hist = getHist(mixed[mixed.Type==2].Energy_keV, bins=bins)
plt.plot(hist[0],hist[1], 'b-', label='Electrons')

hist = getHist(mixed[mixed.Type==3].Energy_keV, bins=bins)
plt.plot(hist[0],hist[1], 'g-', label='Photons')

plt.legend()
plt.xlabel('Energy [$keV$]')
plt.ylabel('Counts [$-$]')
plt.grid()
plt.show()
```

3 Particle Identification Analysis

```
import numbers, glob, os, re
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
from scipy import ndimage
from matplotlib.colors import LogNorm
import analyseDose as ad
import keras
from keras import optimizers
```

```
from matplotlib import pyplot as plt
from numpy.random import seed
from tensorflow import set_random_seed
from keras.preprocessing.image import Imagedata_generator
from keras.models import Sequential
from keras.layers import Dropout, Flatten, Dense
from keras import applications
import os



def get_meas_info(paths):



    Parameters:
        paths (string / list of strings)
    '''
        for path in paths:
        path=os.path.abspath(path)
        fileNames.append(path)

        match=re.search('E([\d]+)_D([\d.]+)_C([\+-][\d.]+)_I
           ([\+-][\d.]+)_([\d]+)ms', path)
        if match:
            energy_MeV.append(float(match.group(1)))
            depth_mm.append(float(match.group(2)))
            crossline_mm.append(float(match.group(3)))
            inline_mm.append(float(match.group(4)))
```

```
            acq_time_ms.append(int(match.group(5)))

    # returns a dictionary with absolute paths and
       measurements details
    return {'path': fileNames[0] if len(fileNams)==1 else
       fileNames,
            'energy_MeV':energy_MeV[0] if len(energy_MeV)==1
                else energy_MeV,
            'depth_mm':depth_mm[0] if len(depth_mm)==1 else
               depth_mm,
            'crossline_mm':crossline_mm[0] if len(
               crossline_mm)==1 else crossline_mm,
            'inline_mm':inline_mm[0] if len(inline_mm)==1
               else inline_mm,
            'acq_time_ms':acq_time_ms[0] if len(acq_time_ms)
               ==1 else acq_time_ms}


def read_clusters_elist(paths,
                        sensor_size_mm = np.array([14.08,
                           14.08, 0.3]),
                        sensor_size_px = np.array([256, 256,
                            1]),
                        filters = True):


    Parameters:
```

```
    paths (str or list)
    sensor_size_mm (array)
    sensor_size_px (array)
'''

# computes detector parameters
pixel_size_um = (sensor_size_mm[0]/sensor_size_px[0])*1
   e3
sensor_thick_um = sensor_size_mm[2]*1e3

data=[]

# reads raw cluster lists and computes cluster
   parameters
for path in paths:

    data_single=pd.read_csv(
        path, sep=';',
        skiprows=2,
        names = ['DetectorID', 'EventID', 'x_mm', 'y_mm
           ', 'Edep_keV', 't_ns', 'Flags', 'Size_px', '
           Height_keV',
                  'BorderPixCount', 'Roundness', '
                     Angle_rad', 'Linearity', 'Length_px
                     ', 'Width_px'])
```

```
dictionary = {t: i for i, t in enumerate(np.sort(
    data_single.t_ns.unique()))}
data_single['frame_no'] = data_single['t_ns'].apply(
    lambda x: dictionary[x])

data_single['CenterX_mm']=-(data_single.x_mm-
    sensor_size_mm[0]/2)
data_single['CenterY_mm']=(data_single.y_mm-
    sensor_size_mm[1]/2)

data_single['Length_um'] = data_single.Length_px *
    pixel_size_um
data_single['Length3D_um'] = np.sqrt(data_single.
    Length_um**2 + (sensor_thick_um)**2)
data_single = data_single[data_single['Length3D_um']
    > 0]
data_single['Width_um'] = data_single.Width_px *
    pixel_size_um

data_single['Beta_deg'] = np.degrees(np.arctan(
    data_single.Length_um/(sensor_size_mm[2]*1e3)))

data_single['LET_keVum'] = data_single.Edep_keV /
    data_single.Length3D_um
data_single = data_single[data_single['LET_keVum'] >
    0]
```

```
        data_single['LETAq_keVum'] = 10**((-0.2902) + 1.025
            * (np.log10(data_single.LET_keVum))) # Benton, E.
             R, 2010

        data_single['EperPx_keV'] = data_single['Edep_keV']
            / data_single['Size_px']

    if len(data)==1:
        data=data[0]
    else:
        data=pd.concat(data)

    return data


def recognize_clusters(clusters_elist,


    '''

    Parmeters:
        clusters_elist (DataFrame)
        display_info (bool)

    '''
    clusters_elist['Edep_px'] = clusters_elist['Edep_keV'] /
        clusters_elist['Size_px']
```

```
EpxCut_Ph = 35
RoundCut_Ph = 0.45
EpxCut_Pr = 35
SizeCut_Pr = 7
SizeCut_El = 2
EpxCut_El = 35
HeightCut_El = 100

clusters_elist.loc[(clusters_elist['Roundness']>0.75) &
   (clusters_elist['Size_px']>SizeCut_El)  & (
   clusters_elist['Size_px']<8)  & (clusters_elist['
   Edep_px']<EpxCut_El) & (clusters_elist['Height_keV']<
   HeightCut_El),'Type_3'] = 1
clusters_elist.loc[(clusters_elist['Size_px']>SizeCut_El
   ) & (clusters_elist['Edep_px']<EpxCut_El) & (
   clusters_elist['Height_keV']<HeightCut_El),'Type_3']
   = 1

for x, idx in enumerate(clusters_elist):
    clusters_elist[idx] = clusters_elist[idx].fillna(0)

clusters_elist['rec_sum'] = clusters_elist['Type_1'] +
   clusters_elist['Type_2'] + clusters_elist['Type_3']
```

```
        clusters_elist.loc[(clusters_elist['Type_1'] == 1) &
            (clusters_elist['Type'] = 'photon'
        clusters_elist.loc[(clusters_elist['Type_2'] == 1) &
            (clusters_elist['Type'] = 'proton'
        clusters_elist.loc[(clusters_elist['Type_3'] == 1) &
            (clusters_elist['Type'] = 'electron'


def get_histogram(data,
                  bins=np.arange(0, 5 + 0.01, 0.01),
                  kind='mean'):

    '''
    Parameters:
        data (iterable object)
        bins (array)
        kind (str): 'sum', 'mean', 'std', 'median', 'min' or
            'max'
    '''



    # convert to DataFrame
    if not isinstance(data, pd.DataFrame):
        data = pd.DataFrame(data)

    # creates a histogram for one set of data
    if data.shape[1] == 1:
```

```
        hist = list(np.histogram(data.iloc[:, 0], bins=bins)
            )

    # creates histogram if two sets of data are provided

def cut_frame(clusters_elist,
              frame_size_mm,
              sensor_size_mm = np.array([14.08, 14.08, 0.3])
                 ):

    '''
    Parameters:
        clusters_elist (DataFrame)
        frame_size_mm (number)
        sensor_size_mm (array)
    '''

    return clusters_elist[~((clusters_elist.CenterX_mm<(-
       sensor_size_mm[0]/2+frame_size_mm)) | (clusters_elist
       .CenterX_mm>(sensor_size_mm[0]/2-frame_size_mm)) | (
       clusters_elist.CenterY_mm<(-sensor_size_mm[1]/2+
       frame_size_mm)) | (clusters_elist.CenterY_mm>(
       sensor_size_mm[1]/2-frame_size_mm)))]



def plot_edep_maps(clusters_elist_path,
```

```
                clusters_log_all_path,
                frame_range=[0,100],
                sensor_size_mm=14.08,
                sensor_size_px=256,
                cond_pos_mm=0.1,
                figsize=[20,8],
                show_figure=True,
                save_figure=False,
                output_file_path='EdepMap.pdf', ):

clusters_elist = read_clusters_elist(clusters_elist_path
    )
recognize_clusters(clusters_elist, display_info=False)

if clusters_elist.isnull().values.any():
    clusters_elist.fillna(value='None', inplace=True)
else:
    pass



imgs_proton=[]
imgs_photon=[]
imgs_electron=[]
cluster_data=[]
pixel_size_mm=sensor_size_mm/sensor_size_px

analysed_clusters=0
```

```
matched_clusters=0
match_dup=0
match_non=0

with open(clusters_log_all_path) as fp:
    for cnt, line in enumerate(fp):
        # read frame number
        if re.findall(r'^Frame', line):
            frame_no=int(re.findall(r'^Frame (\d+)',
               line)[0])
            continue
        if frame_no<frame_range[0]:
            continue
        if frame_no>=frame_range[1]:
            break
        if line=='\n':
            continue

        data_line=re.findall(r'\[(\d+),(\d+),(\d+.\d+),\
           d+\]', line.replace(' ',''))
        data_line=np.array(data_line).astype('float')
        analysed_clusters+=1

        # create img for a single cluster
        img=np.zeros([sensor_size_px,sensor_size_px])
        img[data_line[:,0].astype(int), data_line[:,1].
           astype(int)]=data_line[:,2]
```

```
# calculate morphology
mass_centre_mm=np.array(ndimage.measurements.
    center_of_mass(img))*pixel_size_mm

# generate cluster data
single_cluster_info=pd.DataFrame({'x_mm':
    mass_centre_mm[0], 'y_mm': mass_centre_mm[1],
                                'Edep_keV': [data_line
                                    [:,2].sum()],
                                'Size_px': [data_line
                                    [:,2].shape[0]],
                                'Height_keV': [
                                    data_line[:,2].max
                                    ()],
                                'frame_no': [frame_no
                                    ]})
```

```
            clusters_elist_row=clusters_elist.loc[(
                clusters_elist.x_mm>=(mass_centre_mm[0]-
                cond_pos_mm)) & (clusters_elist.x_mm<=(
                mass_centre_mm[0]+cond_pos_mm)) & (
                clusters_elist.y_mm>=(mass_centre_mm[1]-
                cond_pos_mm)) & (clusters_elist.y_mm<=(
                mass_centre_mm[1]+cond_pos_mm)) & (
                clusters_elist.Size_px==data_line[:,2].shape
                [0]) & (clusters_elist.frame_no==(frame_no+1)
                )]

            if clusters_elist_row.shape[0]>1:
                match_dup+=1
#               print('dupicate (clusters_elist_row shape:
    {:d}, frame_no: {:d})'.format(clusters_elist_row.shape
    [0], frame_no))
                continue

            if clusters_elist_row.shape[0]==0:
                match_non+=1
#               print('empty (clusters_elist_row shape: {:d
    }, frame_no: {:d})'.format(clusters_elist_row.shape[0],
    frame_no))
                continue

            matched_clusters+=1
            # continue when no type in elist
```

```
            if clusters_elist_row.Type.values[0]=='None':
                continue

            # assign to groups of types
            if   clusters_elist_row.Type.values[0]=='proton
               ':
                imgs_proton.append(img)
            elif clusters_elist_row.Type.values[0]=='photon
               ':
                imgs_photon.append(img)
            elif clusters_elist_row.Type.values[0]=='
               electron':
                imgs_electron.append(img)

            single_cluster_info=single_cluster_info.join(
               clusters_elist_row.reset_index(), lsuffix='
               _clu')
            cluster_data.append(single_cluster_info)

    cluster_data=pd.concat(cluster_data).reset_index().drop(
       columns='index')

gs = gridspec.GridSpec(4,1)
fig = plt.figure(figsize=(15, 10))



nom_energy_mev = 150
```

```
direction = 'Deep' # beam profile direction (it is only for
   our convenience when we analyze the beam profiles)
depth_mm = [30, 120, 157, 161]
distance_mm = 37

range_shifter = False # beam range modulator
sensor_thick_um = 300

i = 0
i1 = 1
ax = fig.add_subplot(gs[i])

    # bins=np.linspace(0.1,10,100) # for x-axis as linear
       scale
    bins=np.logspace(-2,2,200) # for x-axis as log scale

    #plt.figure(dpi=1200)
    ### Sim. ###
    # sim - norm. factor
    prim_no=3500000000 # the number of primary protons (from
        the spreadsheet)
    norm_sim=1E8/prim_no # norm. factor for sim. data

    # sim - plot protons
    hist = get_histogram(sim_data[sim_data.ParticleName=='
       proton'].LETAq_keVum, bins=bins)
```

```
ax.plot(hist[0],hist[1]*norm_sim, color = 'tab:cyan',
    drawstyle="default", label='SIM\t$p^+$ ({:.2f}%)'.
    format((sim_data['ParticleName']=='proton').sum()/
    sim_data.shape[0]*100))


### AI ###
# AI - norm. factor
AInorm_proton_no=sim_data[sim_data.ParticleName=='proton
    '].shape[0]*norm_sim # normalized number of prim.
    protons
AInorm_tpx=AInorm_proton_no/AItpx_data[AItpx_data.Type
    =='proton'].shape[0] # norm. factor for measurement
    data


# AI - plot protons
hist = get_histogram(AItpx_data[AItpx_data.Type=='proton
    '].LETAq_keVum, bins=bins)

ax.plot(hist[0],hist[1]*AInorm_tpx, color = 'tab:blue',
    drawstyle="default")
hist_uncer = get_histogram(tpx_data[tpx_data.Type=='
    proton'].LETAq_keVum_UNCER, bins=bins)
```

```
    ax.fill_between(hist[0], hist[1]*AInorm_tpx+tpx_data.
        LETAq_keVum_UNCER[i], hist[1]*AInorm_tpx-tpx_data.
        LETAq_keVum_UNCER[i], color='dimgray')

    ax.fill_between(hist[0], hist[1]-AItpx_data[AItpx_data.
        Type=='proton'].Uncertaity, hist[1]+AItpx_data[
        AItpx_data.Type=='proton'].Uncertaity, color='grey',
        alpha=0.3)

    i = i+1

    ax.set_yscale('log') # set y-axis as log scale
    ax.set_xscale('log') # set x-axis as log scale
    ax.set_ylim([1e0,1e4]) # set y-axis range
    ax.set_xlim([1e-1,2e1]) # set x-axis range, should
        correspond to binning
    plt.grid(True, which="both",linestyle='--',alpha=0.4)



fig.text(0.5, 0.04, 'LET in water [$keV/\mu m$]', ha='center
    ', va='center',fontsize=12)
fig.text(0.06, 0.5, 'Counts norm. to 10$^8$ protons [-]', ha
    ='center', va='center', rotation='vertical',fontsize=12)
fig.savefig('result.pdf', dpi=700)
plt.show()
```

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import glob as glob
from sklearn.neighbors import KernelDensity
import matplotlib.gridspec as gridspec

#matplotlib inline

#subplot
gs = gridspec.GridSpec(4,1)
fig = plt.figure(figsize=(10, 10))

def kernel_approx(bins, mu, sigma, points=1000):
    gauss = np.random.normal(mu, sigma, points)
    kernel, binedge = np.histogram(gauss, bins = bins)
    return [binedge, kernel/points]
def get_kernel_histogram(data, bins,n_sd=3):
    if not isinstance(data, pd.DataFrame):
        data = pd.DataFrame(data)
    histu=np.zeros(len(bins)-1)
    histl=np.zeros(len(bins)-1)
    histm=np.zeros(len(bins)-1)
    for i in data.itertuples(index=False):
        mu=i[data.columns.get_loc('LETAq_keVum')]
        sigma=i[data.columns.get_loc('LETAq_keVum_UNCER')]/3
            #assume this distance covers 3sd range
```

```
        binedge, kernel=kernel_approx(bins,mu,sigma)
        sigs=np.sqrt(i[data.columns.get_loc('Uncertainty')
            ]/100)
        histu=histu+kernel*(1+n_sd*sigs)
        histl=histl+kernel*(1-n_sd*sigs)
        histm=histm+kernel
    bincen = binedge[:-1] + np.diff(bins) / 2
    return [bincen, histu, histl, histm]



def get_histogram(data, bins=np.arange(0, 5 + 0.01, 0.01)):

    if not isinstance(data, pd.DataFrame):
        data = pd.DataFrame(data)

    hist = list(np.histogram(data.iloc[:, 0], bins = bins))
    hist[1] = hist[1][:-1] + np.diff(bins) / 2

    return [hist[1], hist[0]]



# params
nom_energy_mev = 150
direction = 'Deep' # beam profile direction (it is only for
   our convenience when we analyze the beam profiles)
depth_mm = [30, 120, 157, 161]
```

```
distance_mm = 37
range_shifter = False # beam range modulator
sensor_thick_um = 300

i = 0
i1 = 1

for depth in depth_mm:
# read meas. data
    #print(glob.glob('./tpx_data/E{:d}_D{:06.2f}_C*{:06.2f}
        _I*.csv'.format(nom_energy_mev, depth_mm[0],
        distance_mm))[0])
    tpx_data=pd.read_csv(glob.glob('./tpx_data/E{:d}_D{:06.2
        f}_C*{:06.2f}_I*.csv'.format(nom_energy_mev, depth_mm
        [0], distance_mm))[0])

    # read sim. data
    sim_data=pd.read_csv(glob.glob('./sim_data/E{:d}_D{:06.2
        f}_C*{:06.2f}_I+000.00.csv'.format(nom_energy_mev,
        depth_mm[0], distance_mm))[0])

    # read AI data
    AItpx_data=pd.read_csv('./ai_data/AITPX_{:d}MeV_Deep_
        {:06.2f}mm_{:06.2f}mm.csv'.format(nom_energy_mev,
        depth_mm[0], distance_mm))

    # MC sim. - LET in water
```

```
sim_data = sim_data[sim_data['Edep_keV']>0]
sim_data['LETAq_keVum'] = 10**((-0.2902) + 1.025 * (np.
    log10(sim_data.LET_keVum))) # Benton, E. R, 2010



ax = fig.add_subplot(gs[i])

bins=np.logspace(-2,2,200) # for x-axis as log scale

### Sim. ###
# sim - norm. factor
prim_no=3500000000 # the number of primary protons (from
     the spreadsheet)
norm_sim=1E8/prim_no # norm. factor for sim. data

# sim - plot protons
hist = get_histogram(sim_data[sim_data.ParticleName=='
    proton'].LETAq_keVum, bins=bins)
ax.plot(hist[0],hist[1]*norm_sim, color = 'tab:cyan',
    drawstyle="default", label='Simulations')



### AI ###
# AI - norm. factor
AInorm_proton_no=sim_data[sim_data.ParticleName=='proton
    '].shape[0]*norm_sim # normalized number of prim.
    protons
```

```
    AInorm_tpx=AInorm_proton_no/AItpx_data[AItpx_data.Type
        =='proton'].shape[0] # norm. factor for measurement
        data

    # AI - plot protons
    hist = get_histogram(AItpx_data[AItpx_data.Type=='proton
        '].LETAq_keVum, bins=bins)
    ax.plot(hist[0],hist[1]*AInorm_tpx, color = 'tab:blue',
        drawstyle="default", label = 'Measurement')



    #tpx_data_AI=tpx_data.join(AItpx_data.Uncertainty)
    tpx_data_AI=AItpx_data.join(tpx_data.LETAq_keVum_UNCER)
    bincen, hist_uncer_u, hist_uncer_l, hist_uncer_m =
        get_kernel_histogram(tpx_data_AI[tpx_data_AI.Type=='
        proton'], bins,3)
#     ax.plot(bincen,hist_uncer_m*AInorm_tpx, color = 'tab:
    red',linewidth=0.5, drawstyle="default", label = '
    Measurement Uncertain')
    ax.fill_between(bincen, hist_uncer_l*AInorm_tpx,
        hist_uncer_u*AInorm_tpx, color='grey' ,alpha=0.3)
    ############

    i = i+1

    ax.set_yscale('log') # set y-axis as log scale
```

```
    ax.set_xscale('log') # set x-axis as log scale
    ax.set_ylim([1e0,1e4]) # set y-axis range
    ax.set_xlim([3e-1,1.5e1]) # set x-axis range, should
       correspond to binning
    plt.grid(True, which="both",linestyle='--',alpha=0.4)
    ax.set_ylabel('Counts norm. to\n 10$^8$ protons [-]')

ax.set_xlabel('LET in water [$keV/\mu m$]')
#fig.supylabel('Counts norm. to 10$^8$ protons [-]')
#fig.supxlabel('LET in water [$keV/\mu m$]')

# fig.savefig('long_37mm.pdf', bbox_inches = 'tight')
```

BIBLIOGRAPHY

[1] E. Alpaydın, *Introduction to Machine Learning*. MIT.

[2] R. D. Reed and R. J. Marks, *Neural Smithing: Supervised Learning in Feedforward Artificial Neural Networks*. MIT Press, 1999.

[3] B. Vandeginste, D. Massart, L. Buydens, S. De Jong, P. Lewi, and J. Smeyers-Verbeke, "Chapter 44 - artificial neural networks," in *Handbook of Chemometrics and Qualimetrics: Part B*, ser. Data Handling in Science and Technology, B. Vandeginste, D. Massart, L. Buydens, S. De Jong, P. Lewi, and J. Smeyers-Verbeke, Eds. Elsevier, 1998, vol. 20, pp. 649–699. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0922348798800543

[4] S. Vieira, W. Pinaya, and A. Mechelli, "Using deep learning to investigate the neuroimaging correlates of psychiatric and neurological disorders: Methods and applications," *Neuroscience Biobehavioral Reviews*, vol. 74, 01 2017.

[5] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," *Communications of the ACM*, vol. 60, pp. 84 – 90, 2012.

[6] M. Xiao, Y. Wu, G. Zuo, S. Fan, H. Yu, Z. Shaikh, and Z. Wen, "Addressing overfitting problem in deep learning-based solutions for next generation data-driven networks," *Wireless Communications and Mobile Computing*, vol. 2021, pp. 1–10, 08 2021.

[7] S. S. Yadav and S. M. Jadhav, "Deep convolutional neural network based medical image classification for disease diagnosis," *Journal of Big Data*, vol. 6, no. 1, pp. 1–18, 2019.

[8] I. Goodfellow, Y. Bengio, and A. Courville, *Deep Learning*. MIT Press, 2016, http://www.deeplearningbook.org.

[9] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," *arXiv preprint arXiv:1409.1556*, 2014.

[10] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, "Generative adversarial networks," *Communications of the ACM*, vol. 63, no. 11, pp. 139–144, 2020.

[11] M. W. Browne, "Cross-validation methods," *Journal of Mathematical Psychology*, vol. 44, no. 1, pp. 108–132, 2000. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0022249699912798

[12] S. C. Hora, "Aleatory and epistemic uncertainty in probability elicitation with an example from hazardous waste management," *Reliability Engineering System Safety*, vol. 54, no. 2, pp. 217–223, 1996, treatment of Aleatory and Epistemic Uncertainty. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0951832096000774

[13] E. Zio and N. Pedroni, *Uncertainty characterization in risk analysis for decision-making practice*. FonCSI, 2012.

[14] E. Hüllermeier and W. Waegeman, "Aleatoric and epistemic uncertainty in machine learning: an introduction to concepts and methods," *Machine Learning*, vol. 110, no. 3, pp. 457–506, mar 2021. [Online]. Available: https://doi.org/10.1007%2Fs10994-021-05946-3

[15] J. M. Brown and G. Leontidis, "Chapter 11 - deep learning for computer-aided diagnosis in ophthalmology: a review," in *State of the Art in Neural Networks and their Applications*, A. S. El-Baz and J. S. Suri, Eds. Academic Press, 2021, pp. 219–237. [Online]. Available: https://www.sciencedirect.com/science/article/pii/B9780128197400000115

[16] M. Abdar, F. Pourpanah, S. Hussain, D. Rezazadegan, L. Liu, M. Ghavamzadeh, P. Fieguth, X. Cao, A. Khosravi, U. R. Acharya *et al.*, "A review of uncertainty quantification in deep learning: Techniques, applications and challenges," *Information Fusion*, vol. 76, pp. 243–297, 2021.

[17] J. Denker and Y. LeCun, "Transforming neural-net output levels to probability distributions," *Advances in neural information processing systems*, vol. 3, 1990.

[18] J. David, "A practical bayesian framework for backpropagation networks," *Neural computation*, vol. 4, no. 3, pp. 448–472, 1992.

[19] R. M. Neal, *Bayesian learning for neural networks*. Springer Science & Business Media, 2012, vol. 118.

[20] D. A. Nix and A. S. Weigend, "Estimating the mean and variance of the target probability distribution," in *Proceedings of 1994 ieee international conference on neural networks (ICNN'94)*, vol. 1. IEEE, 1994, pp. 55–60.

[21] A. Kendall and Y. Gal, "What uncertainties do we need in bayesian deep learning for computer vision?" *Advances in neural information processing systems*, vol. 30, 2017.

[22] M. A. Hull, C. J. Rees, L. Sharp, and S. Koo, "A risk-stratified approach to colorectal cancer prevention and diagnosis," *Nature Reviews Gastroenterology & Hepatology*, vol. 17, no. 12, pp. 773–780, 2020.

[23] D. K. Rex, C. Kahi, M. O'Brien, T. Levin, H. Pohl, A. Rastogi, L. Burgart, T. Imperiale, U. Ladabaum, J. Cohen, and D. A. Lieberman, "The american society for gastrointestinal endoscopy pivi (preservation and incorporation of valuable endoscopic innovations) on real-time endoscopic assessment of the histology of diminutive colorectal polyps," *Gastrointestinal Endoscopy*, vol. 73, no. 3, pp. 419–422, 2011. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0016510711000253

[24] R. Singh, V. Owen, A. Shonde, P. Kaye, C. Hawkey, and K. Ragunath, "White light endoscopy, narrow band imaging and chromoendoscopy with magnification in diagnosing colorectal neoplasia," *World journal of gastrointestinal endoscopy*, vol. 1, no. 1, p. 45, 2009.

[25] I. I. Vişovan, M. Tanțău, O. Pascu, L. Ciobanu, and A. Tanțău, "The role of narrow band imaging in colorectal polyp detection," *Bosnian Journal of Basic Medical Sciences*, vol. 17, no. 2, p. 152, 2017.

[26] S. Barbeiro, D. Libânio, R. Castro, M. Dinis-Ribeiro, and P. Pimentel-Nunes, "Narrow-band imaging: clinical application in gastrointestinal endoscopy," *GE-Portuguese Journal of Gastroenterology*, vol. 26, no. 1, pp. 40–53, 2019.

[27] S.-E. Kim, H. Y. Paik, H. Yoon, J. E. Lee, N. Kim, and M.-K. Sung, "Sex-and gender-specific disparities in colorectal cancer risk," *World journal of gastroenterology: WJG*, vol. 21, no. 17, p. 5167, 2015.

[28] A. White, D. Joseph, S. H. Rim, C. J. Johnson, M. P. Coleman, and C. Allemani, "Colon cancer survival in the united states by race and stage (2001-2009): findings from the concord-2 study," *Cancer*, vol. 123, pp. 5014–5036, 2017.

[29] D. M. Denbow, "Gastrointestinal anatomy and physiology," in *Sturkie's avian physiology*. Elsevier, 2015, pp. 337–366.

[30] J. B. O. Gautier, W. Peters, and M. B. Huhmann, "Gastrointestinal physiology," 2017.

[31] X. Shen, A. Jain, O. Aladelokun, H. Yan, A. Gilbride, L. M. Ferrucci, L. Lu, S. A. Khan, and C. H. Johnson, "Asparagine, colorectal cancer, and the role of sex, genes, microbes, and diet: A narrative review," *Frontiers in Molecular Biosciences*, vol. 9, 2022.

[32] B. Vogelstein, E. R. Fearon, S. R. Hamilton, S. E. Kern, A. C. Preisinger, M. Leppert, A. M. Smits, and J. L. Bos, "Genetic alterations during colorectal-tumor development," *New England Journal of Medicine*, vol. 319, no. 9, pp. 525–532, 1988.

[33] Y. LeCun, Y. Bengio, and G. Hinton, "Deep learning," *nature*, vol. 521, pp. 436–444, 2015.

[34] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," *CoRR*, vol. abs/1409.1556, 2015.

[35] H. Qassim, A. Verma, and D. Feinzimer, "Compressed residual-vgg16 cnn model for big data places image recognition," in *2018 IEEE 8th Annual Computing and Communication Workshop and Conference (CCWC)*, 2018, pp. 169–175.

[36] C. Teyssier, J. Bouchami, F. Dallaire, J. Idárraga, C. Leroy, S. Pospísil, J. Solc, O. Scallon, and Z. Vykydal, "Performance of the medipix and timepix devices for the recognition of electron-gamma radiation fields," *Nuclear Instruments and Methods in Physics Research Section A: Accelerators, Spectrometers, Detectors and Associated Equipment*, vol. 650, no. 1, pp. 92–97, 2011, international Workshop on Semiconductor Pixel Detectors for Particles and Imaging 2010. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0168900210028949

[37] X. Llopart, R. Ballabriga, M. Campbell, L. Tlustos, and W. Wong, "Timepix, a 65k programmable pixel readout chip for arrival time, energy and/or photon counting measurements," *Nucl. Instrum. Meth. A*, vol. 581, pp. 485–494, 2007, [Erratum: Nucl.Instrum.Meth.A 585, 106–108 (2008)].

[38] C. Granja, K. Kudela, J. Jakubek, P. Krist, D. Chvatil, J. Stursa, and S. Polansky, "Directional detection of charged particles and cosmic rays with the miniaturized radiation camera MiniPIX Timepix," *Nucl. Instrum. Meth. A*, vol. 911, pp. 142–152, 2018.

[39] J. Jakubek, "Precise energy calibration of pixel detector working in time-over-threshold mode," *Nucl. Instrum. Meth. A*, vol. 633, pp. S262–S266, 2011.

[40] D. Bolst, S. Guatelli, L. T. Tran, L. Chartier, M. L. Lerch, N. Matsufuji, and A. B. Rosenfeld, "Correction factors to convert microdosimetry measurements in silicon to tissue in 12C ion therapy," *Physics in Medicine and Biology*, vol. 62, no. 6, pp. 2055–2069, 2017.

[41] L. Chartier, L. T. Tran, D. Bolst, S. Guatelli, A. Pogossov, D. A. Prokopovich, M. I. Reinhard, V. Perevertaylo, S. Anderson, C. Beltran, N. Matsufuji, M. Jackson, and A. B. Rosenfeld, "Microdosimetric applications in proton and heavy ion therapy using silicon microdosimeters," *Radiation Protection Dosimetry*, vol. 180, no. 1-4, pp. 365–371, 2018.

[42] V. A. Pan, F. Pagani, B. James, D. Bolst, S. Peracchi, J. Vohradsky, A. Pogossov, S. Guatelli, M. Petasecca, M. L. Lerch, D. A. Prokopovich, D. Boardman, V. L. Perevertaylo, S. D. Chambers, A. G. Williams, S. Werczynski, T. Inaniwa, N. Matsufuji, M. Jackson, L. T. Tran, and A. B. Rosenfeld, "Characterization of a novel large area microdosimeter system for low dose rate radiation environments," *Nuclear Instruments and Methods in Physics Research, Section A: Accelerators, Spectrometers, Detectors and Associated Equipment*, vol. 1002, no. March, p. 165238, 2021. [Online]. Available: https://doi.org/10.1016/j.nima.2021.165238

[43] D. Bolst, S. Guatelli, L. T. Tran, and A. B. Rosenfeld, "Erratum: The impact of sensitive volume thickness for silicon on insulator microdosimeters in hadron therapy (Phys. Med. Biol. (2020) 65 (035004) DOI: 10.1088/1361-6560/ab623f)," *Physics in Medicine and Biology*, vol. 66, no. 5, 2021.

[44] E. R. Benton, E. V. Benton, and A. L. Frank, "Conversion between different forms of LET," *Radiation Measurements*, vol. 45, no. 8, pp. 957–959, 2010. [Online]. Available: http://dx.doi.org/10.1016/j.radmeas.2010.05.008

[45] M. A. Cortés-Giraldo and A. Carabe, "A critical study of different Monte Carlo scoring methods of dose average linear-energy-transfer maps calculated in voxelized geometries irradiated with clinical proton beams," *Physics in Medicine and Biology*, vol. 60, no. 7, pp. 2645–2669, 2015.

[46] C. Granja, J. Jakubek, S. Polansky, V. Zach, P. Krist, D. Chvatil, J. Stursa, M. Sommer, O. Ploc, S. Kodaira, and M. Martisikova, "Resolving power of pixel detector timepix for wide-range electron, proton and ion detection," *Nuclear Instruments and Methods in Physics Research Section A: Accelerators, Spectrometers, Detectors and Associated Equipment*, vol. 908, pp. 60–71, 2018. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0168900218309586

[47] C. Granja, C. Oancea, J. Jakubek, L. Marek, E. Benton, S. Kodaira, J. Miller, A. Rucinski, J. Gajewski, P. Stasica, V. Zach, J. Stursa, D. Chvatil, and P. Krist, "Wide-range tracking and LET-spectra of energetic light and heavy charged particles," *Nuclear Instruments and Methods in Physics Research, Section A: Accelerators, Spectrometers, Detectors and Associated Equipment*, vol. 988, no. May 2020, p. 164901, 2021. [Online]. Available: https://doi.org/10.1016/j.nima.2020.164901

[48] R. Nabha, O. Van Hoey, C. Granja, A. Parisi, M. De Saint-Hubert, L. Struelens, C. Oancea, E. Sterpin, V. Zach, J. Stursa, A. Rucinski, J. Gajewski, P. Stasica, and F. Vanhavere, "A novel method to assess the incident angle and the let of protons using a compact single-layer timepix detector," *Radiation Physics and Chemistry*, vol. 199, p. 110349, 2022. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0969806X22003929

[49] P. Stasica, J. Baran, J. Gajewski, C. Granja, C. Oancea, M. Pawlik-Niedźwiecka, M. Rydygier, A. Schavi, and A. Ruciński, "Oc-0576: Timepix for characterization of mixed radiation field produced in proton radiotherapy," *Radiotherapy and Oncology*, vol. 152, pp. S323–S324, 2020.

[50] G. Kalyani and B. Janakiramaiah, "Chapter 11 deep learning-based detection and classification of adenocarcinoma cell nuclei," in *Trends in Deep Learning Methodologies*, ser. Hybrid Computational Intelligence for Pattern Analysis, V. Piuri, S. Raj, A. Genovese, and R. Srivastava, Eds. Academic Press, 2021, pp. 241–264. [Online]. Available: https://www.sciencedirect.com/science/article/pii/B9780128222263000118

[51] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, "Generative adversarial nets," in *Advances in Neural Information Processing Systems*, Z. Ghahramani, M. Welling, C. Cortes, N. Lawrence, and K. Weinberger, Eds., vol. 27. Curran Associates, Inc., 2014. [Online]. Available: https://proceedings.neurips.cc/paper/2014/file/5ca3e9b122f61f8f06494c97b1afccf3-Paper.pdf

[52] T. Salimans, I. Goodfellow, W. Zaremba, V. Cheung, A. Radford, and X. Chen, "Improved techniques for training gans," 2016. [Online]. Available: https://arxiv.org/abs/1606.03498

[53] L. Mescheder, A. Geiger, and S. Nowozin, "Which training methods for gans do actually converge?" 2018. [Online]. Available: https://arxiv.org/abs/1801.04406

[54] A. Radford, L. Metz, and S. Chintala, "Unsupervised representation learning with deep convolutional generative adversarial networks," 2015. [Online]. Available: https://arxiv.org/abs/1511.06434

[55] Z. Chen, M. Jin, Y. Deng, J.-S. Wang, H. Huang, X. Deng, and C.-M. Huang, "Improvement of a deep learning algorithm for total electron content maps: Image completion," *Journal of Geophysical Research: Space Physics*, vol. 124, no. 1, pp. 790–800, 2019.

[56] L. Trippa, L. Waldron, C. Huttenhower, and G. Parmigiani, "Bayesian nonparametric cross-study validation of prediction methods," *The Annals of Applied Statistics*, vol. 9, no. 1, pp. 402 – 428, 2015. [Online]. Available: https://doi.org/10.1214/14-AOAS798

[57] T. Fawcett, "An introduction to roc analysis," *Pattern Recogn. Lett.*, vol. 27, no. 8, p. 861–874, jun 2006. [Online]. Available: https://doi.org/10.1016/j.patrec.2005.10.010

[58] E. Hüllermeier and W. Waegeman, "Aleatoric and epistemic uncertainty in machine learning: An introduction to concepts and methods," *Machine Learning*, vol. 110, no. 3, pp. 457–506, 2021.

[59] T. Duerig, A. Pelton, and D. Stöckel, "An overview of nitinol medical applications," *Materials Science and Engineering: A*, vol. 273-275, pp. 149–160, 1999. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0921509399002944

[60] B. O'Brien, F. Weafer, and M. Bruzzi, "1.3 shape memory alloys for use in medicine," *Comprehensive Biomaterials II; Ducheyne, P., Ed.; Elsevier: Oxford, UK*, pp. 50–78, 2017.

[61] L. M. Bidaut, J. L. Humm, G. S. Mageras, and L. N. Rothenberg, "8 - imaging in radiation oncology**this chapter is an update and expansion of material presented in the first edition by c.c. ling, r. mohan, l.e. reinstein, and l.n. rothenberg, and in the second edition by the current authors." in *Leibel and Phillips Textbook of Radiation Oncology (Third Edition)*, third edition ed., R. T. Hoppe, T. L. Phillips, and M. Roach, Eds. Philadelphia: W.B. Saunders, 2010, pp. 120–154. [Online]. Available: https://www.sciencedirect.com/science/article/pii/B9781416058977000081

[62] L. J. Olafsen, B. Jones, L. Sparks, H. H. Nguyen, A. Tanner, K. E. Schubert, J. S. Olafsen, S. Dayawansa, E. Fonkem, and J. H. Huang, "Current-controlled nitinol wire for improved arterial navigation," in *Advanced Biomedical and Clinical Diagnostic and Surgical Guidance Systems XVII*, vol. 10868. SPIE, 2019, pp. 94–100.

[63] R. O. Bude and R. S. Adler, "An easily made, low-cost, tissue-like ultrasound phantom material," 1995.

[64] R. Nicholson and M. Crofton, "Training phantom for ultrasound guided biopsy." *The British journal of radiology*, vol. 70, no. 830, pp. 192–194, 1997.

[65] M. Buda, A. Maki, and M. A. Mazurowski, "A systematic study of the class imbalance problem in convolutional neural networks," *Neural Networks*, vol. 106, pp. 249–259, 2018. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0893608018302107