Fall 2021

Two-Level Data Augmentation with Transfer Learning for Classification of Medical Images with Limited Data

Nihil Pudota
San Jose State University

TWO-LEVEL DATA AUGMENTATION WITH TRANSFER LEARNING FOR CLASSIFICATION OF MEDICAL IMAGES WITH LIMITED DATA

A Thesis

Presented to

The Faculty of the Department of Electrical Engineering

San José State University

In Partial Fulfillment

of the Requirements for the Degree

Master of Science

by

Nihil Pudota

December 2021

The Designated Thesis Committee Approves the Thesis Titled

TWO-LEVEL DATA AUGMENTATION WITH TRANSFER LEARNING
FOR CLASSIFICATION OF MEDICAL IMAGES WITH LIMITED DATA

by

Nihil Pudota

APPROVED FOR THE DEPARTMENT OF ELECTRICAL ENGINEERING

SAN JOSÉ STATE UNIVERSITY

December 2021

Birsen Sirkeci, Ph.D.	Department of Electrical Engineering
Chang Choo, Ph.D.	Department of Electrical Engineering
Leonard Wesley, Ph.D.	Department of Computer Science

ABSTRACT

TWO-LEVEL DATA AUGMENTATION WITH TRANSFER LEARNING FOR CLASSIFICATION OF MEDICAL IMAGES WITH LIMITED DATA

by Nihil Pudota

Machine learning used in the medical industry can potentially detect cancer in human cells at an early stage. However, training the machine learning models, especially deep learning models require thousands to millions of samples in order to reach an acceptable accuracy level. It is well-know that obtaining medical data is tedious hence in most cases, medical datasets have limited number of data samples. One solution for this problem is utilizing transfer learning such as pretrained networks on another dataset. Another solution is to increase the number of training data points with data augmentation. Common data augmentation methods for images include not only simple techniques such as transforming images using rotation and flipping, but also generative adversarial networks (GANs). However, one critical question is “Does the original dataset have enough to train a GAN?”. In most scenarios, the answer is “No” for this critical question. In this thesis, we propose a two-level data augmentation technique (simple data augmentation based on image transformations followed by a GAN) with transfer learning, which is tested on a small dataset of cancer cell images. The dataset used in this research consists of lung and colon cancer samples, each containing different types of cancers. Only part of the original dataset is used for experimenting in order to mimic small dataset environment. Our results show that the proposed method is able to achieve an accuracy of 94.1% even when 150 original images used for training. This is very close to 97.33% accuracy achieved if one uses all the available training data which is 12000 samples.

DEDICATION

I would like to dedicate this work to my parents. Without their support and sacrifice I would not have been able to achieve everything that I have done in my life. I love you both and appreciate everything that you have provided and done for me. This thesis is also dedicated to my sister who could not wait for me to finish. I will move out soon, but you know where to find me.

ACKNOWLEDGEMENTS

I would like to thank my advisor Dr. Sirkeci for her guidance and upmost patience throughout this research. Without her support and mentorship this research would not have been possible. Her flexibility and understanding throughout the COVID-19 time allowed me to accomplish my goals in a timely and efficient manner. I would like to share my gratitude and appreciation for her support. I would also like to thank Dr. Choo and Dr. Wesley for their time and effort in reviewing all the work that I have put forth.

Lastly, I would like to thank my friends and family for their immense support while writing my thesis. A special shoutout to my girlfriend, Alisha for her endless support during this process. I promise I'll hangout more soon.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

LIST OF ABBREVIATIONS

GAN - Generative Adversarial Networks
CNN - Convolutional Neural Network
ILSVRC - ImageNet Large Scale Visual Recognition Challenge
WGAN - Wasserstein GAN
BEGAN - Boundary Equilibrium Generative Adversarial Networks
SEGAN - Speech Enhancement Generative Adversarial Network
OPD - Optical path delay
Lung_n – Lung benign tissue
Lung_aca – Lung adenocarcinoma
Lung_scc – Lung squamous cell carcinoma
Colon_n – Colon benign tissue
Colon_aca – Colon adenocarcinoma
KNN - K-nearest neighbor
FID - Frechet Inception Distance
TOP-GAN - Transferring of a Pre-trained generative adversarial network
HIPAA - Healthy Insurance Portability and Accountability Act
ANN - Artificial Neural Networks
SVM - Support Vector Machines

1 INTRODUCTION

The human race is plagued with many diseases that take the lives of many worldwide. Most of these diseases can be mitigated or cured either with medicine or vaccines. Cancer is one of the world's biggest public health concerns and is the second leading cause of death in the United States [1]. With no cure or vaccine, cancer has the ability to attack any age group regardless of previous health. Lung and bronchus cancer had the highest number of deaths in the United States in both males and females in 2020 [1]. Figure 1 shows the breakdown of the estimated death by sex in the United States for the year 2020. Data is obtained from [1].

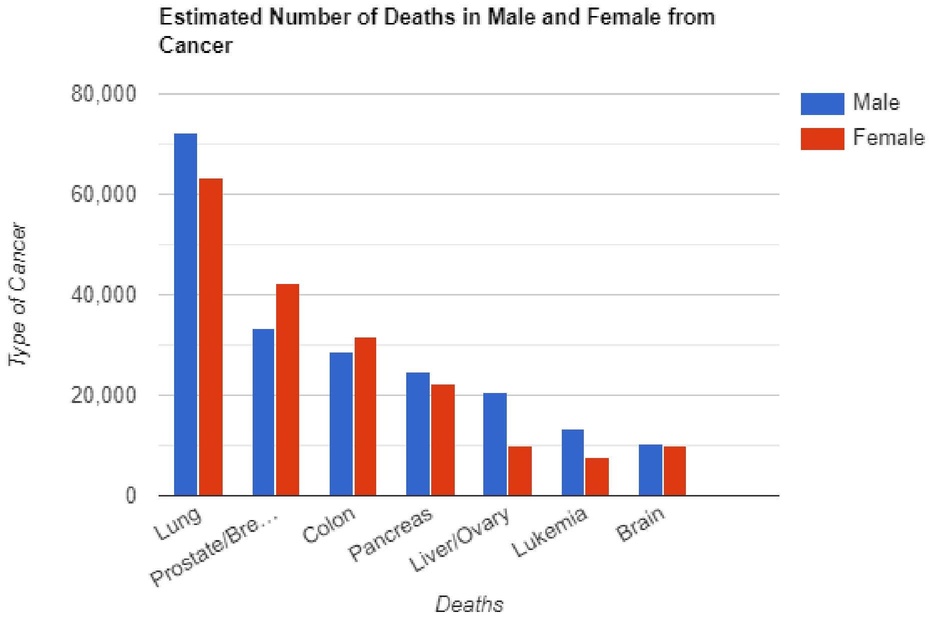

Fig. 1. Top leading cancer types for estimated number of deaths in 2020 in the United States.

Cancer is such a deadly and prevalent disease worldwide; however, if detected early, it can be mitigated and treated. Though not fully curable, early detection can help cancer not be fatal. A controlled investigation was performed on men 40 and over with lung cancer in [2]. These men were split into two groups, those who received 6-month chest radiographs over three years and those who did not have access to x-rays. The research showed that the 5-year survival rate for the average population was 15%, but in the group that received x-rays, the survival rate was 23%, and the survival rate was only 6% for those who did not receive any x-rays [2]. The same pattern existed for the survival rate for squamous carcinoma and adenocarcinoma, which showed survival rates of 28% and 25% with x-ray facilitates compared to the 15% and 0% survival rates respectively. The research concluded that with earlier detection of lung cancer, we can improve survival rates [2]. The diagnosis of cancer can be automated with machine learning and deep learning.

Machine learning can learn and understand from previous historical examples and patterns. The ability to break down complex datasets makes machine learning a good contender for cancer prognosis and prediction [3]. Machine learning is a developing technique for cancer prognosis and detection as papers published on this topic have increased over five hundred percent from 1994 to 2005 [3]. Various methods were attempted to determine the best model for cancer detection. It was found that the bottleneck in most of these works was the size and complexity of a given training set [4]. With limited data, any model is prone to overfitting and reported extremely high accuracies, misleading the model’s actual performance [3]. In a recent 2017 study, a deep convolutional neural network was developed in order to detect lung cancer [5]. The study used a dataset that consisted of three different diagnoses of lung cancer that include adenocarcinoma, squamous cell carcinoma, and small cell carcinoma. Having access to only a small dataset that is unable to train an accurate model, data augmentation was done through rotations, flipping, and filtering. This data augmentation was used to prevent the issue of overfitting that occurs with small datasets [5]. The dataset consisted of microscopic images of the three various lung cancers. The implemented deep convolutional neural network obtained a 71% accuracy in classifying the images correctly [5].

There are various approaches to deal with limited dataset problems. Simpler models tend to work better on small training datasets as they are less prone to overfitting. Simpler models, though are unable to extract detailed features, can lead to a higher misclassification rate not yielding a true accuracy. Transfer learning is a technique commonly used for small dataset problems [6]. Transfer learning is the process of training a model on a larger set of images such as the ImageNet dataset, which includes over 14 million images and applying those parameters to the smaller dataset. Transfer learning was used to tackle the detection of diabetic retinopathy. Diabetic retinopathy affects the vision of type 1 and type 2 diabetic patients, but early detection of this can prevent long-term vision impairment [7]. Weights and layers from the Inception-V3 pre-trained models were transferred to the Diabetic retinopathy problem with an addition of a soft-max layer at the output. The weights for all the layers except the output layer are frozen since the model is pre-trained. Previous research on the same problem yielded an accuracy of 87% using 35000 images. The proposed model using transfer learning yielded an accuracy of 90.9%

using 2500 images [7], showing the effectiveness of transfer learning on a small dataset.

This thesis proposes a two-layer data augmentation technique to handle small datasets. The two-layer data augmentation technique combines both simple data argumentations and generative adversarial networks (GAN). The performance of this technique will be compared to that of simple data augmentations and GANs applied separately with and without transfer learning. Chapter 2 will cover some basics on generative adversarial networks, their current developments using convolutional neural networks (CNNs), and transfer learning with pre-trained models. Chapter 3 will introduce some recent literature that performed machine learning techniques on cancer cell data. Chapter 4 will briefly discuss the dataset that was used in the thesis. Chapter 5 will discuss the implemented data augmentation techniques and transfer learning techniques. Next, chapter 6 will discuss the obtained resulted and compare the various techniques. Lastly, Chapter 7 will discuss the conclusion drawn from this research.

2 BACKGROUND

This chapter will first introduce convolutional neural networks as they are crucial to understanding GANs. The structure of CNNs and their architecture will be discussed with some real-world applications. After CNNs, the chapter will discuss the implementation of GANs and their various applications in data augmentation. Lastly, real-world applications and current uses of GAN’s will be discussed in order to show their effectiveness in other use cases.

2.1 Convolutional Neural Networks

Convolutional neural networks have increased in popularity over the last decade when dealing with specific applications that include: image classification, speech recognition, and natural language processing. Convolutional layers are used to extract features from images, and the more layers a CNN has, the more detailed features get extracted in deeper layers [8]. Each layer is composed of multiple neurons. At each neuron, a dot product of the inputs to the neuron and the weights is followed by a nonlinear activation function. Backpropagation is an efficient algorithm for training neural networks, which is the process of estimating these weights in the network.

Multiple different types of layers make up a CNN such as convolutional layers, activation layers, and down-sampling layers. In CNN, an input is fed into a convolutional layer which is a set of learnable kernels that are extracting features from a given input. The output of convolutional layer is created by sliding a filter over the input and performing a convolution operation. The result of this yields features maps at each layer extracting the relevant parts of the input images [8]. When setting up a convolutional layer, the stride and filter size can be controlled as well. The stride and filter size determine the size of the output of the feature maps. For example, if the convolution layer was set to a filter size (4 x 4) with a stride of 2 and an image size of (64 x 64), the output is as follows below:

$$Output \ = \ [\frac{64-4}{2}] + 1 \ = \ 31 \tag{1}$$

The convolution layer is followed by an activation function. The ReLU function is usually the preferred activation function since the training happens much faster relative to the other functions [9]. Different activation functions include Tanh, sigmoid, leaky ReLU, and exponential LU. These activation functions are used in order to provide some non-linearity which lets the model learn more complex patterns [8]. Down-sampling is also called pooling layer. The pooling layer is used to reduce the resolution of the feature maps in order to introduce a small amount of variance into the model. A pooling layer can be either a max-pooling layer or an average-based pooling layer. This is used to down-sample the inputs and decrease the number of parameters which will help with the issue of overfitting in CNNs [8], [9].

After multiple interlaced convolutional and pooling layers, CNNs are accompanied by one or two fully connected layers at the end of the network. The key benefits of using CNNs are due to its success in equivalent representations, sparse interactions, and parameter sharing [10]. Sparse interactions are implemented using kernels that are smaller than the input size. Parameter sharing reduces storage as the model will not need to learn from a separate set of parameters for every location [10]. Equivalent representations ensure that if the input image is translated, the representations also follow the same pattern.

We highlighted how a convolutional neural network is implemented and the various layers which it uses to determine the output. CNNs can identify important features of a data set without any human interventions and have been applied to the real-world applications of computer vision, speech processing, face recognition, and more [10].

2.2 Transfer Learning Using CNN-Based Models

Transfer learning is a technique used primarily when the user does not have access to a large dataset. Transfer learning is a powerful technique that uses model parameters learned while training on one dataset for another dataset in a similar domain. In particular, for deep CNNs, the initial network layer biases and weights are responsible for feature extraction; and hence utilizing initial layers trained for a large dataset for some other problem brings advantage in the form of decreased training time, performance gain, and computational cost. There are cases in real-world applications where getting a large dataset is not possible, in these cases a pre-trained model, which is a model trained on a similar but large dataset can be utilized [11]. Rubin et al, used transfer learning to cope with a small training dataset. They trained a machine learning model on sperm cells and applied that trained model to predict if a skin cell was cancerous or not [11]. They received higher accuracy than classic methods of extending the dataset such as data augmentation [11].

Models such as VGG16 and GoogLeNet are examples of transfer learning and are known as pre-trained networks using ImageNet database [12]. The GoogLeNet network was the winner of the ImageNet Large-Scale Visual Recognition Challenge (ILSVRC) in 2014 [12]. The GoogLeNet network is 22 layers deep, and the authors estimated that the network would converge within less epochs using a few high-end GPUs [13]. The challenge in ILSVRC was to classify 100,000 images in the testing dataset and 50,000 images in the validation dataset that belong to 1000 different classes. Training a neural network using such a large dataset will allow successful extraction of various features for image classification problems [13]. These kinds of pre-trained networks can be used as a baseline for other image classification problems after finetuning using the relevant dataset even if it is limited in size.

The VGG16 is another pre-trained model that is proposed by Simonyan and Zisserman from the University of Oxford [12]. VGG16 is a convolutional neural network that was able to achieve a 92.7% accuracy in ImageNet. The dataset used to train this network is part of ImageNet and is split into three sets: 1.3 million training images, 50 thousand validation images, and 100 thousand testing images [12]. This is a pre-trained model that is easy to access and is known to work with many other image classification problems [12]. The architecture of the VGG16 model involves 41 layers in total including convolution, max pooling, and fully connected layers. It uses both the ReLU and softmax activation functions in the convolution layers and the output layers, respectively. The original input layer is a 224x224 RGB image that is passed through multiple convolution layers. These layers are accompanied by the ReLU activation layer and a max-pooling layer in between the convolution layers [12]. After the stack on the convolution layers three fully connected layers with different channels are connected. The last layer is a softmax layer which makes a prediction of the class of the image.

2.3 Generative Adversarial Networks

Generative Adversarial Networks (GANs) are used to generate and create more data from an existing dataset. GANs are together with supervised, semi-supervised and unsupervised learning methods [14]. GANs consist of both a generator network and a discriminator network. As the name of the networks suggests, the generator's job is to generate images and the discriminator's job is to determine if these are real or fake images. Both networks are trained at the same time and essentially compete with each other [14]. Both the generator and discriminator networks are usually multi-layer neural networks such as CNNs. A random noise is used as input to a generator network which then outputs sample data. The discriminator is fed in the real and fake data sample and is trained to understand the fake data from the real data. The generator is aimed to eventually create images that the discriminator is unable to distinguish, making the fake image very similar to the original. Using this newly generated fake image, one can expand the original limited-size. dataset (i.ie data augmentation) for each class separately.

There are two conflicting goals that need to be met while training a GAN. The discriminator's goal is to maximize its classification accuracy while the generator's goal is to maximally confuse the discriminator [14]. The training loss function captures both aspects and makes the generator and discriminator network compete.

The theoretical optimal solution to this competition exists when the generator is perfect, and the discriminator can no longer tell the difference between the real and fake inputs [15]. A GAN is usually unstable and cannot be trained to the optimal network and will have an early stopping condition that is met [14].

Many different GAN variants were created after the initial structure and logic were in place. One example of a GAN variant is the Wasserstein GAN (WGAN) which was proposed to tackle the vanishing gradient problem [15]. The WGAN is improved and progressed toward more stability while training GANs, but still failed to converge in certain scenarios [15]. Another GAN known as the LS-GAN aimed to improve how the parameters were learned while keeping the original GAN structure [15]. Other GANs were created with further research, and some changed the original GAN structure while some optimized certain loss functions to achieve better results. Some other GAN variants are WGAN, LS-GAN, Semi-GAN, C-GAN, and BiGAN [15]. Each GAN was created in an attempt to solve a certain problem or optimize parameters. GANs opened up a new possibility for new machine learning applications that were not possible with previous techniques.

2.3.1 Applications of Generative Adversarial Networks

GANs are used to generate and augment samples that are very similar to the real samples in the original dataset. GANs can be used when there isn’t sufficient data that can be used effectively for training. In recent years GANs have been popular in image classification and speech recognition [15]. One of the most familiar GAN examples is the generation of faces. Boundary Equilibrium Generative Adversarial Networks (BEGAN) focused on image generation with high visual quality and resolution [16].

GANs not only has applications in the image classification department but also in speech recognition and language processing. Speech Enhancement Generative Adversarial Network (SEGAN) is a GAN that attempts to solve the problem of speech enhancement. The goal of this network is to provide a quick enhancement process for speech signals. It works with raw audio and learns from different speakers and noise types and incorporates them together [17]. SEGAN is able to enhance speech in order to improve the overall quality of the listeners.

Overall, GANs are able to generate data that can easily be interpreted and increase the scope of datasets [15]. This generation of realistic data opens up for the implementation of other machine learning models such as K-nearest neighbor (KNN), CNN, and others which require a large dataset for effective results. GANs are problematic when trying to optimize both the networks at the same time, as there can be cases when the networks will not converge generating unusable images [15]. This thesis focuses on image classification techniques that utilize GANs in order to increase the size of the training dataset.

3 LITERATURE REVIEW

Machine learning was greatly researched in the field of cancer cell classification. A combination of GAN and transfer learning was applied to cancer cell classification [11], deep CNNs were used to classify lung cancer types [5], and hyperspectral colon cancer images were classified using CNNs [18]. A naive Bayes classifier and a K-nearest neighbor classifier were used to classify breast cancer with a 97% accuracy [19]. The various research shows the versatility and effectiveness of machine learning techniques in the medical industry. The models, though cannot yield a 100% accuracy working with human doctors, greatly increase efficiency in cancer detection. With medical data being complex and not as abundant, the bottleneck for high accuracy is small usable training data. In the next subsection, we will look at recent research done in cancer cell classification with the use of GANs and transfer learning. Three research papers will be discussed. The first two are proposing a different method of using GANs for cancer cell detection. At first, a pathologyGAN will be discussed which is a new GAN for pathological images. Then, a hybrid approach using both transfer learning and GAN will be discussed. Finally, we will discuss how transfer learning was used to detect leukemia in blood cells.

3.1 PathologyGAN for Cancer Tissue

The PathologyGAN research aimed to develop a framework that allows GANs to capture key features for tissues in order to generate samples in the latent space. Quiros et al used two cancer tissue datasets in order to develop a model that is able to generate high-quality images [20]. This work was done in order to better understand tumors and their diverse structures and features. In order to evaluate the fake samples, the researchers used the Frechet Inception Distance (FID) metric which is the norm when quantifying a GAN’s performance [20]. To determine the FID distance, feature samples are fitted to Gaussian distributions for both the real and generated features and the difference between the distances are measured. The research also uses expert pathologists in order to determine any significant difference between the generated samples and the real ones. The pathologyGAN uses samples of breast cancer tissue and colorectal cancer to generate images for the respective datasets. The PathologyGAN model is able to generate cancer tissue images that are not distinguishable by expert pathologists. The generative model captures multiple features that are projected into a latent space in order to create realistic generated images [20].

3.2 TOP-GAN Cancer Cell Classification

This research is aimed to perform classification on healthy and cancer cells of both primary and metastatic cancer cells. The machine learning methods in this research attempt to tackle the problem of the small training set. The proposed method is called transferring of a pre-trained generative adversarial network (TOP-GAN). This method combines both transfer learning and GANs to make up for the lack of data in the training set [11]. Rubin et al [11] utilize transfer learning, essentially training the GAN on one type of data and implementing this information for another type of dataset. The paper uses unclassified sperm cell images and classified cancer and non-cancer cell images.

First, the GAN is trained on unlabeled optical path delay (OPD) images of human sperm cells and learns the feature which makes up the OPD images of these biological cells. This discriminator network that is not trained on the sperm cells gets its last layer switched and a classifier is created based on the architecture and knowledge of the discriminator trained on sperm cells. The output now of the discriminator will signify whether the cell is of a healthy skin cell or a cancer skin cell [11]. As a result of this, the research combined both transfer learning in which the sperm cells built the discriminator network and the GAN to determine the decision of healthy and cancer cells. In order to train this TOP-GAN network, an Adam optimizer was used with a learning rate of 1e-5 and beta parameters (exponential decay rates) as follows: β_1 = 0.6, β_2 = 0.99. This network is trained for 900 epochs or until convergence [11]. The result of this network is calculated with different training sets with various sizes and methods. As the number of images in the training set decreases the accuracy difference for the TOP-GAN implementation and a basic CNN implementation increases. The TOP-GAN method is able to cope with a situation where there is a small training dataset, given the access to a set of other biological cells [11].

3.3 Transfer Learning

Loey et al, researched the effectiveness of using deep transfer learning to diagnose Leukemia in blood cells. Leukemia is a deadly disease that takes the lives of many, but survival rates can be increased with early detection [21]. Transfer learning was applied using the AlexNet pre-trained CNN. The experiment was conducted with a 2820 image dataset. The research proposed two different methods of transfer learning both using the AlexNet pre-trained CNN [21].

In the first proposed model, images were fixed to 227x227 size, and data augmentation was performed using translations, reflections, and rotations to extend the dataset. After the images were pre-processed, the Alexnet network was used for feature extraction. At the final output layer, four classifiers were used in order to determine if the cell has been affected by leukemia. These classifiers included support vector machines, linear discriminants, decision trees, and K-nearest neighbors [21]. The accuracy for the listed classifiers is 99.3%, 98.51%, 95.82%, and 99.04%, respectively.

In the second proposed model, images were pre-processed the same as previously stated. Then images were trained over the Alexnet pre-trained network to not only extract features from the data but also classify the images as well [21]. There were no extra classifiers that were used to predict the output. The last three layers of the Alexnet CNN were replaced to be tuned to the leukemia classification problem [21]. The rest of the layers have frozen weights that will not change. This model yielded 100% accuracy in detecting leukemia in blood cells. In both proposed models, the classification problem makes use of transfer learning to have deeper features extracted even with a small dataset. This makes transfer learning an effective technique on small dataset machine learning problems.

4 LUNG/COLON CANCER HISTOPATHOLOGICAL DATASET

This chapter will discuss the dataset used in this thesis. First, we discuss how the lung cancer images were obtained, and next, we discuss the data augmentation that was performed to extend the dataset.

Borkowski et al created the lung and colon cancer histopathological image dataset in order to provide more readily available datasets for the field of machine learning [22]. This dataset is split into both lung and colon cancer cell tissue images. The set contains five different classes which include: lung benign tissue (lung_n), lung adenocarcinoma (lung_aca), lung squamous cell carcinoma (lung_scc), colon adenocarcinoma (colon_aca), and colon benign tissue (colon_n). The original dataset consists of 750 lung images and 250 colon images. These images are all Healthy Insurance Portability and Accountability Act (HIPAA) compliant and validate in total 1250 images (250 images in each class). These images were obtained using a Lecia Microscope MC190 HD camera with the resolution capped at 1024x768 [23]. Figure 2 shows various images from the classes in the dataset.

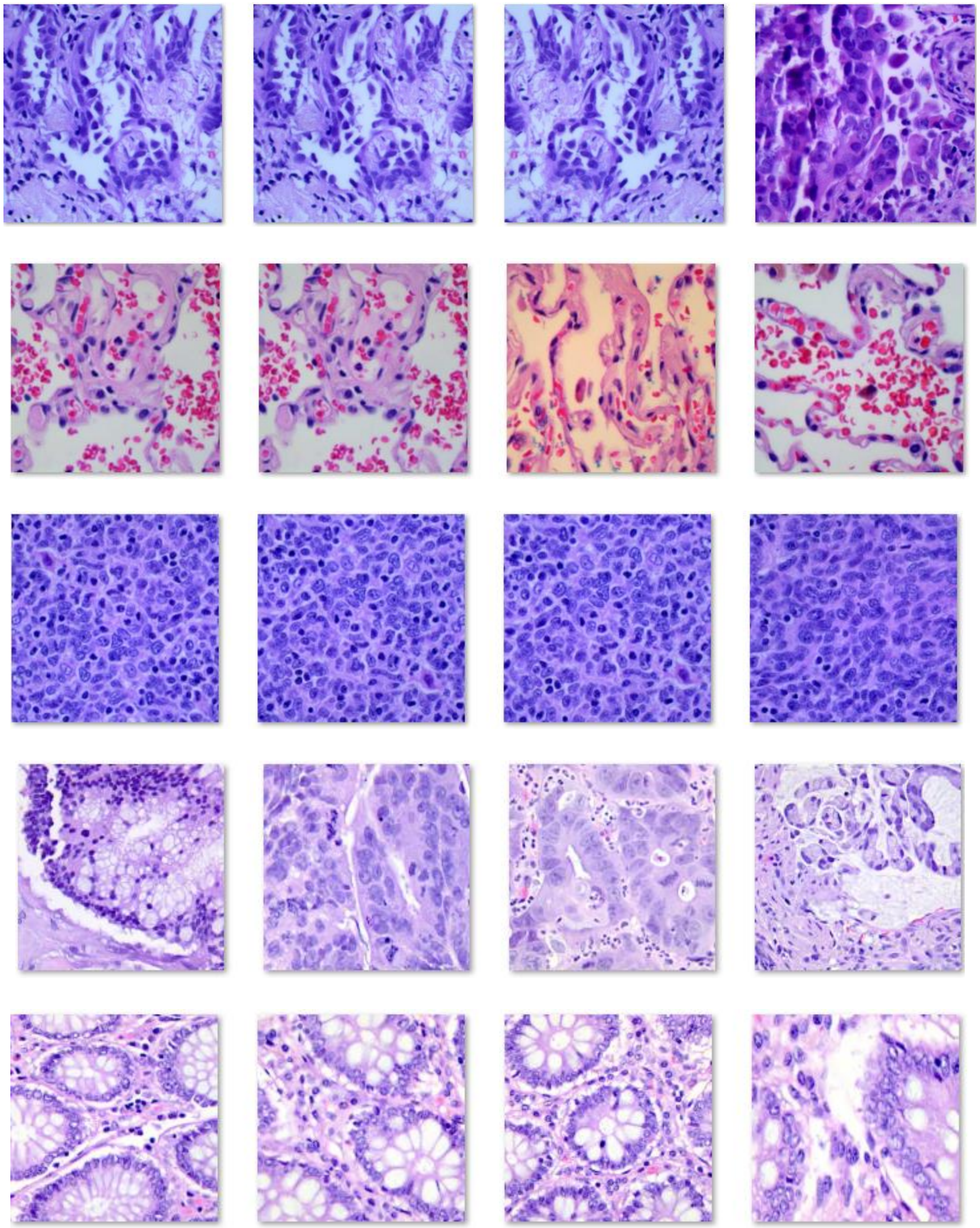

Fig. 2. Four images from each class top to bottom: Lung adenocarcinoma, lung normal tissue, lung squamous cell carcinoma, colon adenocarcinoma, colon normal tissue.

All the images in the dataset are cropped to 768x768 pixels from their original 1024x768 pixels. Next in order to increase the amount of data in the dataset, the images were augmented using the Augmentor library in python [22]. Augmentor is a software package, which is able to generate artificial data based on existing observations, and available in Python and in Julia. This package provides methods such as random rotations, transforms, cropping, zooming, scaling, and resizing in various different angles [24]. The Augmentor was used to expand the lung and colon cancer data as follows: left and right rotations, and horizontal and vertical flips. These data augmentation techniques were able to increase the size of the data set from the original 1250 images to 25000 images. The new augmented dataset now consists of 5,000 images in each class with all images 768x768 pixels in size and in jpeg file format [22]. Figure 2 shows sample images from the lung cancer classes from the dataset.

5 PROPOSED METHOD: A TWO-LEVEL DATA AUGMENTATION TECHNIQUE WITH TRANSFER LEARNING

The method proposed in this research is a two-level data augmentation technique with transfer learning. We will be combining two different data augmentation techniques first, and then a pre-trained network is used for prediction. The first data augmentation technique, discussed in further detail in this section, will be simple data augmentation techniques such as rotations and flips. Following this will be a GAN model that will further generate more images to be used during training. After augmenting the training dataset, the images are fed into the VGG16 pretrained network to make a prediction. The output layer of the VGG16 model will be trained, and all preceding layers will be frozen so the weights will not update. The final or output layer will consist of various machine learning strategies as discussed further below. Figure 3 shows a visual representation of the flow of the proposed method.

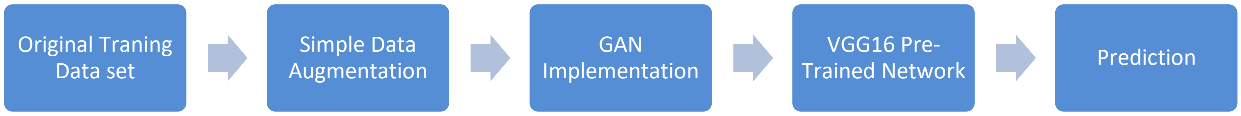

Fig. 3. Visual representation of the proposed method.

Starting with the original dataset, one simple augmentation technique is picked, followed by GAN and VGG16. Finally, one output layer is picked before making a prediction.

5.1 Data Preprocessing for VGG16

The pre-processing of the data before we implement the VGG16 model involves shuffling the dataset and normalizing the images into 384x384 pixels. Shuffling the data ensures that there is some form of randomness in the models and the image size is reduced to enhance computational power and fit to the first layer of VGG16. This is performed before the data is inputted into any of the models discussed in this thesis. Looking at Figure 3 this happened after the GAN implementation and before the VGG16 model in implemented.

We will be using the same VGG16 model described in Chapter 2. This is to research the effectiveness of one- or two-level data augmentation techniques on histopathological data using a small training set. In order to mimic the small training set, we will use partial dataset with a predetermined accuracy level. This small training dataset will be put to test with multiple approaches and will be a standard to compare the effect of various augmentation techniques.

5.2 Output Layer for VGG16

In order to test the effectiveness of the generated images, the dataset is put through a few machines learning models to determine test accuracies. These machine learning model will replace the output layer for the VGG16 network. The following subsections will summarize the various machine learning models used and describe their basic implementations.

5.2.1 Architecture of a Shallow Neural Network

Shallow Artificial Neural Networks (ANNs) are neural networks with only few number of layers. Chapter 2 of this thesis discusses the architecture and application of CNNs, which is an ANN with specialized layers such as convolution and pooling layers. ANNs are used to solve problems involving pattern recognition, clustering, and prediction. In addition to CNNs, ANNs can also come in the forms: feed-forward networks and recurrent networks [25]. Feed-forward networks, as the name suggests has no feedback. Recurrent networks connect back into previous nodes in the network creating a loop, which is feedback. Recurrent networks are usually used with sequential data such as text and speech.

The shallow ANN that will be used in this paper is a simple neural network that will allow us to make a prediction after extracting the features using VGG16. The details about the architecture of the ANN can be seen in Table 1. The first layer is a dense layer with ReLU activation function. The first layer is followed by a batch normalization and dropout layers. There are 4 dense layers that follow the dropout layer with 64, 32, 16, 8 output sizes respectively and with ReLU activation function. The output layer is dense and has a softmax activation function and makes a prediction on the class of the data passed in.

5.2.2 K-Nearest Neighbor

The K-Nearest Neighbor (KNN) algorithm is a supervised learning algorithm that makes predictions based on clusters. The clusters are formed based on the

Table 1. ANN Architecture

Layer	Parameters
Input- Dense	128, activation= relu
Batch Normalization	N/A
Dropout	Frequency = 0.2
Dense	64 activation = relu
Dense	32 activation = relu
Dense	16 activation = relu
Dense	8 activation = relu
Dense	activation = softmax

Euclidean distances from the data points to their neighbors. The amount of neighbors that decide the prediction of a particular data point is known as K [26].

The K value is a hyperparameter and can be optimized on the test set. After the value of K is chosen the algorithm will iterate through all the points in the dataset. For each test point, the Euclidean distance will be found to all other data points and K closest points will be used for prediction: (i) if the problem is classification, the test data point will be assigned a class that is the majority in the K closest points; (ii) if the problem is regression, the prediction for the test data point will be the average of the outputs of the K closest points.

5.2.3 Support Vector Machines

Support Vector Machines (SVMs) are supervised machine learning algorithms that are used for both classification and regression. Linear SVMs aim to create a hyperplane that is able to separate the data into two classes. Figure 4 shows a

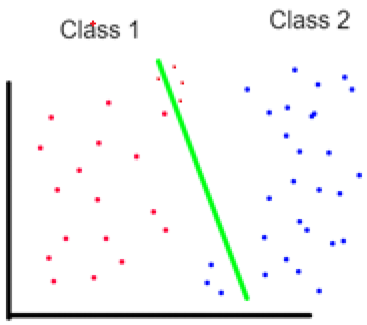

Fig. 4. Visual representation of SVM.

visualization of how the SVM classifies data. Nonlinear SVMs use a kernel to map the data into a different domain and uses a hyperplane in this new domain to separate the dataset into two classes. SVMs can be extended to multiclass problems using various techniques such as one-versus-all or one-vs-one approaches [27], [28].

5.2.4 Random Forest

Random Forest (RF) is also a supervised learning algorithm that uses decision trees for either classification or regression. Multiple decision trees are formed using the same dataset and bagging and the decision from multiple trees are combined using either majority rule (classification) or averaging (regression). While forming the decision trees, the inputs are selected from a random set of possibilities at each node. This is the reason why the algorithm name contains random. RF will not overfit the model and can be used when the data set is high dimensional. When implementing an RF, the most important parameters are the number of trees in the forest and the number of inputs selected at each node. The number of trees in the forest will be equal to the number of outputs that the RF will gather before combining the results of these trees.

5.3 Simple Data Augmentation Techniques

In order to extend the dataset, we propose implementing various different simple data augmentation techniques that will augment the images in the original training set. These techniques include 180-degree rotations, 45-degree rotations, 90-degree rotations, horizontal flips, vertical flips, shear, and blur [29]. These rotations and flips are done to show the image in a new perspective to the machine learning model. Shearing an image will move one part of the image in one direction while moving the other part of the image in the opposite direction. This will distort the image over the horizontal axis. Blurring an image will distort the image slightly while keeping the main structure of the image intact. Adding blue to the dataset will ensure that if there are any distortions in the validation sets, they can be accounted for as well.

5.4 Implementation of Generative Adversarial Networks

The GAN used in this thesis is implemented through TensorFlow and Keras. The GAN implemented is influenced by the research done by Radford et al [30]. The resolution was set to 3 which reduced the images by a factor of 3 as we have limited access to GPUs and had to use a cloud-based notebook that limited the memory usage. This means that we will be working with 96-pixel square images. Next, all the images were pre-processed, sized appropriately to the resolution, and appended into a NumPy array. Once all the images have been loaded and preprocessed into a NumPy array we can build both the discriminator and generator models.

The generator model is a convolutional neural network of which details are given in Table 2. The input layer is a dense layer with a ReLU activation function that takes in the vector size as the input dimension. In this case, the vector size is set to 100. Following this layer is a reshape layer that shapes the input into 4*4*256. Next, we have three batches with the same implementation. First, there is an up-sampling layer followed by a two-dimensional convolutional layer. The convolutional layer is followed by a batch normalization layer and a ReLU activation function. The up-sampling layer is a simple layer with no weight that will double the dimensions of the input. The batch normalization layer keeps the mean close to 0 and the standard deviation as close to 1 as possible using a batch of data at each neuron. These two layers will help keep the generator more stable. After three batches of these layers, there is a final convolutional layer followed by a tanh activation function. The generator model is responsible for creating fake images that the discriminator will need to classify into fake or real.

The discriminator model is also a convolutional neural network like that of the generator. The input layer for the discriminator is a conv2d layer with a leaky ReLU activation function followed by a dropout layer. Next, a convolution layer, batch normalization, and leaky ReLU are repeated three times. The dropout layer randomly sets input units to 0 and this is done to prevent overfitting of a CNN. A leaky ReLU has a small slope for negative input values and is said to be a more robust activation function. After the repeated layers, there is a flattening layer followed by a dense layer with a sigmoid activation function as the output layer.

Table 2. Generator Architecture

Layer	Parameters
Dense	ReLU, Input Dim: 100
Reshape	4*4*256
UpSampling2D	N/A
Conv2D	256 Kernel= 3 Padding = “Same”
BatchNormalization	Momentum = 0.8
Activation	ReLU
Conv2D	256 Kernel= 3 Padding = “Same”
BatchNormalization	Momentum = 0.8
Activation	ReLU
Conv2D	128 Kernel= 3 Padding = “Same”
BatchNormalization	Momentum = 0.8
Activation	ReLU
Conv 2D	3 Kernel = 3 Padding = “Same”
Activation	TanH

The flattening layer will flatten the input shape to a simple vector and that will be consumed by the output layer. The sigmoid activation function in the last layer will t output a value between 0 and 1 which will determine the discriminator’s decision. Table 3 describes the architecture of the discriminator.

Table 3. Discriminator Architecture

Layer	Parameters
Conv2D	32, Kernel = 3, Stride = 2
Activation	Leaky ReLU(0.2)
Dropout	0.25
Conv2D	64, Kernel = 3, Stride = 2
BatchNormalization	Momentum = 0.8
Activation	Leaky ReLU(0.2)
Dropout	0.25
Conv2D	128, Kernel = 3, Stride = 2
BatchNormalization	Momentum = 0.8
Activation	Leaky ReLU(0.2)
Dropout	0.25
Conv2D	256, Kernel = 3, Stride = 1
BatchNormalization	Momentum = 0.8
Activation	Leaky ReLU(0.2)
Dropout	0.25
Conv2D	512, Kernel = 3, Stride = 1
BatchNormalization	Momentum = 0.8
Activation	Leaky ReLU(0.2)
DropOut	0.25
Flatten	N/A
Dense	1, Sigmoid

After building both the architectures for the discriminator and generator we must optimize them using an optimizer. We used the Adam optimization for both the discriminator and the generator. The Adam optimization is picked due to its stochastic gradient descent nature and its computational efficiency. This optimizer is known to do well on most neural networks [31]. The generator loss and the discriminator look are both determined using the Keras binary cross-entropy. This method calculates the cross-entropy loss between the true labels and the predicted labels. Lastly, the learning rate is set to 1.5e-4. Once the generator is trained for a particular number of epochs or till convergence the model (the generator and its weights) will be saved as an .h5 file.

6 EXPERIMENTAL RESULTS

The experimental results of this paper will be split into three different sections. The first section will discuss the results of a standard CNN model, followed by the implementation of the VGG16 model. The second section uses various classifiers as output layer while utilizing VGG16 for feature selection. Then the effect of simple data augmentation will be shown with various models and their accuracies. Next, GANs are implemented to increase the training data set and the change in accuracy is explored. The proposed two-level data augmentation technique is implemented combining GANs and the simple data augmentation techniques. Lastly, we will propose another transfer learning technique combined with simple data augmentation which would classify colon and lung images.

6.1 Basic Model: Standard CNN

The first method that is implemented is a CNN. The architecture of the CNN can be seen in Table 4. It is a 14-layer network that is trained using the lung image dataset. The training was repeated multiple times increasing the number of data points in the original dataset. A fixed test set, 3000 images, was used to validate the dataset for all the runs.

Table 5 shows the validation accuracies of the CNN model using a different number of data points. The model was run over 10 epochs with an Adam optimizer with learning rate of 0.0001. The validation accuracy for the model was unstable over multiple epochs due to the small dataset, and the accuracies were averaged over multiple runs. We can clearly see in Table 5 that the validation accuracy

Table 4. CNN Architecture

Layer	Parameters
Convolution 2D	Filters=32, Kernel=5, activation=relu
MaxPool2D	Pool=2, Stride=2
Convolution 2D	Filters=64, Kernel=5, activation=relu
MaxPool2D	Pool=2, Stride=2
Convolution 2D	Filters=128, Kernel=5, activation=relu
MaxPool2D	Pool=2, Stride=2
Convolution 2D	Filters=256, Kernel=5, activation=relu
MaxPool2D	Pool=2, Stride=2
Dropout	Freq = 0.5
Convolution 2D	Filters=512, Kernel=5, activation=relu
MaxPool2D	Pool=2, Stride=2
Dropout	Freq = 0.5
Flatten	N/A
Dense	Filters=3, activation=softmax

Table 5. CNN Validation Accuracy with No Data Augmentation

Number of Training Data Points	Validation Accuracy
75	65.9%
150	72.2%
300	76.7%
750	82.6%

increases as the number of data points in the training set increases. Note that the same test set was utilized for all these runs.

6.2 VGG16 Without Data Augmentation

Transfer learning helps combat the problem of a small dataset. The VGG16 is a pre-trained neural network that can be used to extract the relevant features of our lung dataset. The extracted features from VGG16 are stored and fed to an RF classifier, which is the best performing model among the among the one proposed in Section 5.2, with a consistent random seed and 50 trees. This last step in the model will predict the class and provide us with the training and validation accuracies for the dataset. This is iterated multiple times while changing the number of data samples in the training lung cancer dataset. The results can be found in Table 6. As expected, as the number of training samples increases, validation accuracy gets better.

Table 6. VGG16/RF Accuracies on Different Number of Original Training Data

Number of Training Data Points	Validation Accuracy
75	81.9%
150	85.1%
300	87.6%
750	90.2%

The ANN, KNN, SVM, and RF algorithms, all described in detail in Chapter 5, will be used as output layers for a variety of different training data sizes to understand the effect on the validation accuracies. There will be 3000 images separated as the validation set, which will not be seen by any of the algorithms during training. The models will be evaluated over three different parameters: validation accuracy, recall, and F1-score. The validation accuracy is the accuracy of the model on the data set that is not used for training. The recall is the amount of correctly labeled data points that are the true positive from the validation set. The F1- score is the harmonic mean of the precision and recall, the closer to 1 the better the model is performing. The performance of the various models can be seen in Table 7. It is apparent here that with the small number of data points used the validation accuracies are not high and have some randomness in the results.

6.3 VGG16 with One-Level Data Augmentation

6.3.1 Original Dataset

The original data set with simple data augmentation contained 1250 images of lung cancer data. As explained in Chapter 4, the given dataset has a larger augmented dataset of 12000 images for the lung dataset with equal number of images representing every class using left, right rotations and vertical, horizontal flips.

Table 8 and Figure 5 show the validation accuracies using augmented dataset from the VGG16 pre-trained network. Table 9 shows the various different output layers connected to VGG16 and their accuracies in combination with simple data augmentation. Using a 12000-image augmented dataset, we achieve an accuracy of 95.6% on lung cancer classification.

Table 7. Accuracy for Various Machine Learning Methods Using Different Training Datasets

Model	Validation Accuracy	F1 Score	Recall
300 Data Points			
ANN	77.33%	81.25%	78.00%
KNN	83.33%	83.33%	83.33%
SVM	82.00%	86.23%	87.95%
RF	88.67%	89.77%	90.00%
150 Data Points			
ANN	77.66%	79.70%	82.45%
KNN	85.33%	85.46%	85.33%
SVM	84.66%	77.30%	77.33%
RF	88.67%	87.90%	87.98%
75 Data Points			
ANN	75.33%	75.29%	75.33%
KNN	84.66%	86.54%	86.78%
SVM	82.00%	87.61%	88.00%
RF	90.00%	86.71%	87.39%

Table 8*.* VGG16/RF Accuracies on Different Number of Augmented Training Data

Training Data Points	Validation Accuracy
1500	91.3%
3000	93.1%
4500	94.3%
12000	95.6%

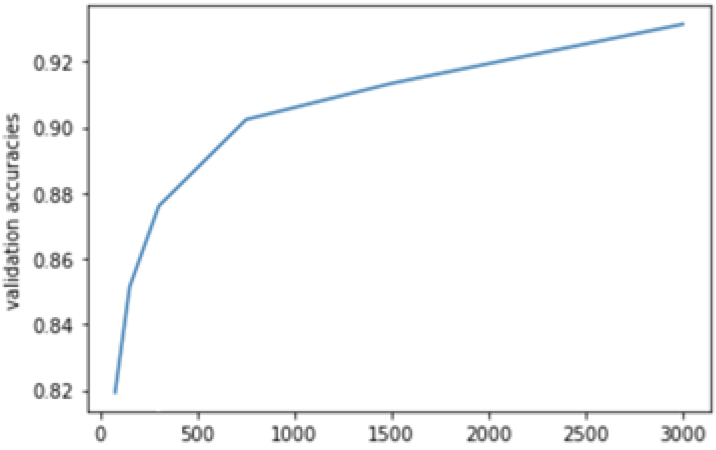

Fig. 5. VGG16 Network: Validation accuracy vs size of the training set.

Table 9. Accuracy Using Augmented Training Dataset for Various Machine Learning Methods

Model	Validation Accuracy	F1 Score	Recall
12000 Data Points			
ANN	92.93%	92.30%	92.27%
KNN	94.87%	94.13%	94.13%
SVM	97.33%	97.13%	97.13%
RF	94.80%	94.01%	94.00%

6.3.2 Simple Data Augmentation

In order to simulate the problem of a small training dataset to show the effectiveness of data augmentation we used up to only 600 images from the original dataset. Data augmentation was used to increase the amount of training data that was available to train the machine learning model from. First, we tested a one-level technique that used various simple data augmentation techniques. The data augmentation techniques, explained in Chapter 5, included rotating, flipping, shearing, and blurring the dataset. Different amounts of training data were used to look at the effects of these simple augmentation techniques.

Table 10 shows the effect of various simple data augmentation techniques on the validation accuracies of the model. We can easily conclude that in any augmentation technique the number of data points available is correlated with the validation accuracy. Shear was the outlier augmentation technique that distorted the image and hurt the accuracy of the model. Rotations, flips, and blurs all increased validation accuracy as the training dataset increased in size. The table shows the least effective and most effective simple data augmentation techniques.

6.3.3 GAN Implementation

This section discusses the generated images produced by the GAN. The architecture of the GAN includes that of the discriminator and generator which can be found in Chapter 5. The goal of the GAN is to generate images that cannot be distinguished from the original dataset. This theoretically will augment and create more data points in the training data to increase the accuracy of our machine learning algorithms. In order to train the GAN, we first start with a latent space vector of random noise shown in Figure 6.

Starting with random noise, the generator will learn over multiple epochs various features from the original dataset which will help create better-looking images. Figure 7 shows the generated images over multiple epochs with a training set of 500 randomly picked images from the entire datasets.

Table 10. Results for Simple Data Augmentation Techniques

Data Augmentation	Training Data	300	375	450	525	600	1200
180 Degree Rotations	ANN	85.23	88.16	93.12	93.61	91.74	94.62
	KNN	73.77	79.1	79.5	77	78.49	71.638
	SVM	90.2	91.4	92.3	92.91	92.971	94.56
	RF	91.28	91.8	91.59	92.362	92.81	92.71
90 Degree Rotations	ANN	90.64	92.49	93.28	90.45	93.81	94.65
	KNN	73.7	78.9	79.17	78.86	78.681	71.652
	SVM	90.203	91.1	92.36	93.16	92.812	94.493
	RF	91.275	92.16	92.61	92.16	92.507	92.667
45 Degree Rotations	ANN	89.1	93.6	92.4	92.5	93.4	93.4
	KNN	73.7	78.4	79.1	78.6	78.16	71.6
	SVM	90.2	91.5	92.3	93.2	92.81	94.4
	RF	91.3	92.2	92.7	92.1	92.5	92.7
Horizontal	ANN	85.1	89.13	92.46	93.42	93.29	94.29
	KNN	72.71	78.94	79.12	76.85	78.681	71.652
	SVM	89.67	91.058	92.362	93.16	92.812	94.493
	RF	90.12	92.1	92.61	92.16	92.51	92.667
Vertical	ANN	93.33	94.203	93.4	94.51	94.91	95.22
	KNN	73.77	78.9	79.1	76.85	78.61	71.652
	SVM	90.2	91.1	92.36	93.16	92.812	94.52
	RF	91.23	92.159	92.61	92.16	92.51	92.67
Shear	ANN	83.1	86.88	87.61	73.7	75.3	82.19
	KNN	56.5	43.1	42.4	40.8	36.9	37.04
	SVM	55.42	33.6	33.13	48.8	47.9	53.089
	RF	70.841	74.3	71.59	72.1	77.8	82.6
Blur	ANN	93.6	93.3	89.91	93.8	94.2	93.841
	KNN	73.7	78.9	77.5	76.5	78.61	78.3
	SVM	90.2	91.1	92.3	93.2	92.81	93.8
	RF	91.3	91.2	90.8	92.2	92.5	91.2

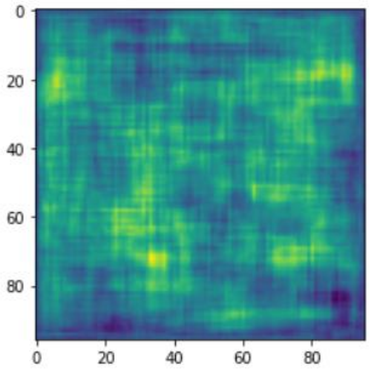

Fig. 6. Random noise of vector size 100x100.

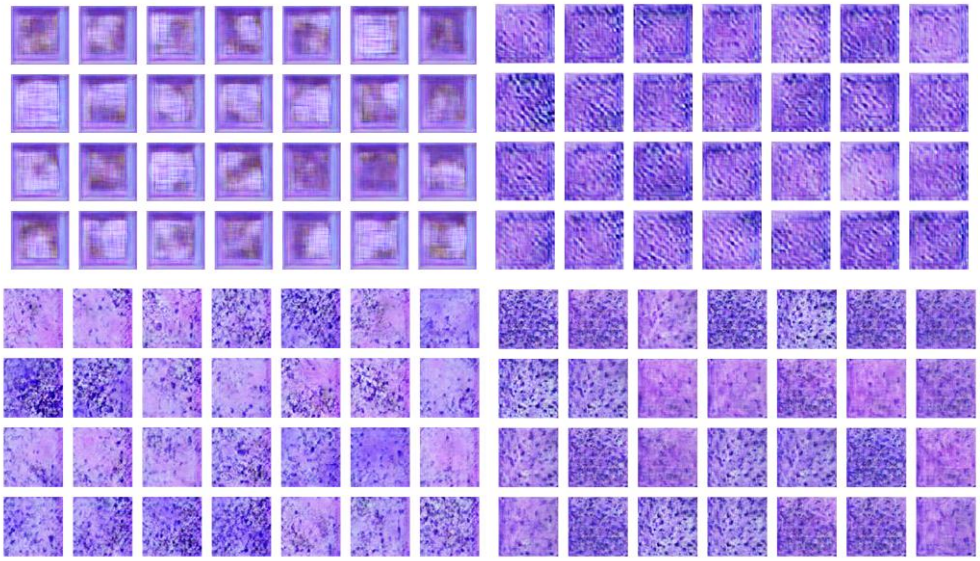

Fig. 7. Generated Lung SCC images, 20 epochs (top left), 50 epochs (top right), 200 epochs (bottom left), 450 epochs (bottom right).

Looking at Figure 6, it is visually recognizable that the latent vector is learning and getting more detailed as the number of epochs increases. Both the generator and discriminator in this model are optimized on the Adam optimizer with a learning rate of 1.5e-4 and a β_1 value of 0.5. After multiple tests, we recognize that at 500 epochs the training seems to plateau, and the images did not get much better. The GAN is run twice to generate images with a training data size of 250 per class and a training data size of 100 per class. This was done to understand the effect of a larger training sample on the generated images.

Once the model was trained, its weights and biases were then stored into a .h5 file so the model can be loaded without needing to be trained again. Six models were generated, three with a training size of 100 and three with a training data size of 250. Figure 8 shows the difference between the original dataset and the generated images from two different GAN models.

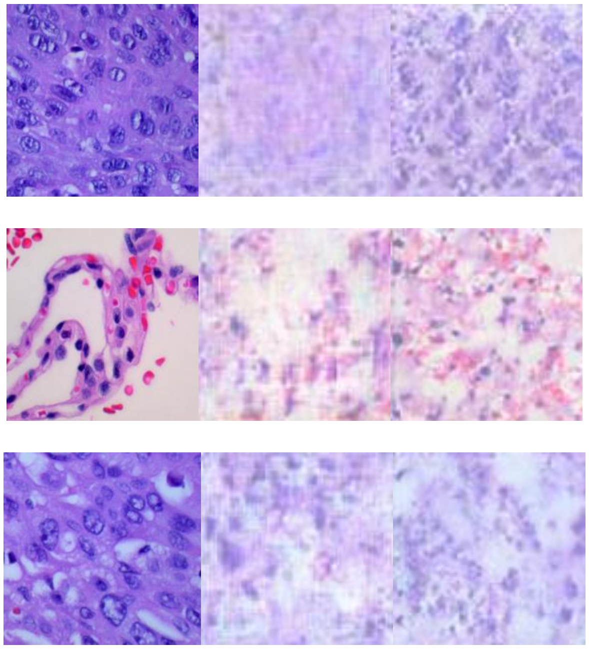

Fig. 8. Lung squamous cell carcinoma (top), Lung benign (middle), Lung adenocarcinoma (bottom), Original Image (left), GAN with 100 data(middle), GAN with 500 data (right).

After generating the images, the best way to test the performance of algorithms using these images was to compare it to the accuracies obtained in the previous section.

6.3.4 Comparing Models Using Generated Images with GAN

This section will discuss the accuracies obtained from various machine learning techniques using the newly generated data from the data set. Two runs were performed with the two sets of generated images from different GAN models. In order to have a controlled setting, for all models, the training data from the original data set will be the same and the validation set will also be the same. The machine learning algorithms will exhibit the same parameters and will not change between both these runs. There will be 100 data points that are appended to 100 generated images per class. Table 11 shows the accuracies for the various models with the generated images from the GAN using 100 data points per class and Table 12 shows the accuracies for the GAN using 250 data points per class.

Table 11. Model Accuracy for Images Trained with 100 Original Data Points Augmented with 100 Generated Data Points Per Class

Model	Validation Accuracy	F1 Score	Recall
ANN	88.67%	90.535%	90.67%
KNN	80.00%	82.79%	82.67%
SVM	80.25%	84.67%	84.11%
RF	87.33%	87.76%	88.00%

As seen in Tables 11 and 12, we observe an increase in accuracies when there were more images in the training dataset. The model that gains most due to additional data is ANN, which is expected due to its complexity compared to the rest of the methods [22]. When compared to the previous accuracies using 300 data points (see Table 7), we see that KNN, SVM, and RF perform worse on the first set

Table 12. Model Accuracy for Images with 100 Original Data Points Augmented with 250 Generated Data Points Per Class

Model	Validation Accuracy	F1 Score	Recall
ANN	93.33%	92.02%	92.00%
KNN	83.33%	84.70%	84.67%
SVM	80.67%	82.44%	83.33%
RF	89.33%	89.75%	90.00%

of generated images (see Table 11). However, if we increase the number of generated data points for each class, we observe KNN, and RF performance get better with a high margin (see Table 12). The results in Table 11 and Table 12 are obtained by averaging over 10 different runs, which helps stabilize the values especially for ANN.

6.4 Proposed Method: VGG16 with Two-Level Data Augmentation

In the previous two subsections, the methods of various simple data augmentation techniques and GANs were tested. This thesis proposes to utilize a two-level data augmentation method, which is sequentially using simple data augmentation followed by GANs (see Figure 3). The main motivation behind this is due to the need that GANs also require a good amount of data to be properly trained. By utilizing simple data augmentation methods such as rotation, flipping, etc, a small dataset can be increased to a data size which is enough to successfully train a GAN. This way the data is augmented further using GANs, which brings in additional gains in performance. Below are our experimentations with the two-level data augmentation.

Training images will be added using simple data augmentations followed by a GAN to double the size of the training dataset. Initially, we use 150 original images from the dataset to mimic small datasets. These 150 images were augmented using a simple technique such as flipping to increase the training dataset to 300 images. This newly created dataset is run through a GAN to create a final dataset consisting of 600 images. The results for these methods can be seen in Table 13. The results are an average of 10 different runs with the same validation set. The validation accuracy of the two-level data augmentation is significantly better than the validation accuracies of one-level techniques described in the Sections 6.3.2 and 6.3.4.

Table 13. Results on Two-Step Data Augmentation

Output Layer of Vgg16	180	45	90	Blur	Horizontal	Shear	Vertical
ANN	91.7	92.783	92.8	93.37	91	91.35	94.1
KNN	53.7	37.7	65.3	54.6	55.13	54.7	52.8
SVM	90.1	89.4	90.7	89.6	89.1	89.2	90.8
RF	90.3	89.88	89.7	90.2	90.913	90.58	90.5

In order to understand the gains from two-level data augmentation, we present a performance summary of all the models discussed in this thesis in Table 14. When we compare the performance of similar training sets, the gains due to two-level data augmentation is highly significant (see last two rows in Table 14). Figure 9 shows the ROC curves for each of the lung classes in our dataset. ROC curves are used for binary classifiers; however, for multiclass problems, we can focus on one class at a time, which is the positive class, and obtain the curve assuming the rest of the classes represent the negative class. Each curve plots the true positive rate (i.e.,

Table 14. Results from Implemented Models

Model	Data Augmentations	Training Dataset	Validation Accuracy
CNN	No	750	82.6%
VGG16 + RF	No	750	90.2%
VGG16 + ANN	No	300	77.33%
VGG16 + SVM	No	300	82.00%
VGG16 + KNN	No	300	83.33%
VGG16 + RF	Yes (simple)	12000	95.60%
VGG16 + ANN	Yes (simple)	12000	92.83%
VGG16 + SVM	Yes (simple)	12000	97.33%
VGG16 + KNN	Yes (simple)	12000	94.87%
VGG16 + RF	Yes (GAN)	600	89.33%
VGG16 + ANN	Yes (GAN)	600	93.83%
VGG16 + SVM	Yes (GAN)	600	80.67%
VGG16 + KNN	Yes (GAN)	600	83.33%
VGG16 + ANN	Yes (Simple Horizontal)	300	85.11%
VGG16 + ANN	2-Level (Vertical+ GAN)	600	94.11%
VGG16 + ANN	2-Level (Blur + GAN)	600	93.37%

recall) as a function of false positive rate (false positives / total number of actual negatives). The proposed two-level data augmentation works better for lung adenocarcinoma class than for lung squamous cell carcinoma. We have also added ROC curve for disease (lung adenocarcinoma or lung squamous cell carcinoma) versus normal.

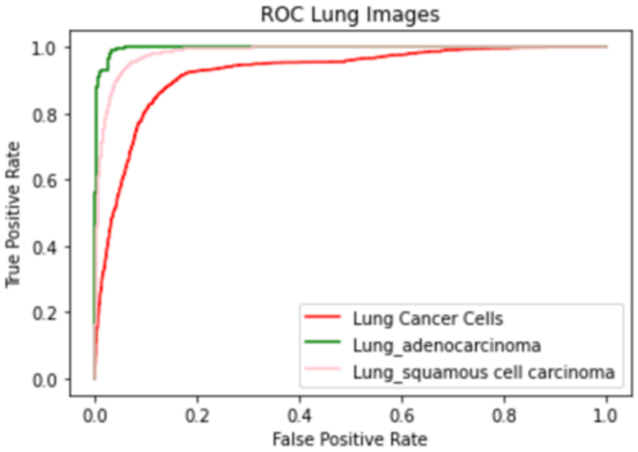

Fig. 9. ROC curve of lung classes with the proposed method.

6.5 Replacing VGG16 with Pre-trained CNN

Previously, we used the VGG16, a pre-trained model using ImageNet, followed by an output layer trained on the lung dataset. This is a method of transfer learning using a pre-trained network. Implementing a similar technique, we propose to create our own pre-trained network using a CNN architecture that will be trained on the colon dataset and be used on a small batch of the lung images. Chapter 4 describes both the colon and lung dataset. The CNN architecture is shown in Table 4 in Section 6.1. The CNN is first trained over the entire colon image dataset. These weights are saved and will not be trained or changed again. The last output layer is replaced with a dense layer that takes in 3 filters and is a softmax activation function as the lung images have three classes. The last layer is trained and the model uses the previous weights trained by the colon dataset in order to predict classes for the lung dataset.

This transfers the features learned from the colon dataset to the lung images. In total, 300 lung images are used for training the output layer and 3000 images are used to validate the model. We receive an accuracy that ranged from 84% to 86% with an average accuracy of 85%. This simulates a situation where we are presented with a large number of images of one type of cell but are lacking images of the other type of cell.

We have also repeated the experiment under the assumption that the colon dataset was small in size, but we have enough lung cell samples. Similarly, transfer learning is applied such that the features learned from the lung dataset are utilized for classifying colon cell images. Note that we modify the output layer of the CNN architecture in order to accommodate the fact that the colon dataset had only two classes. The pre-trained model is able to predict colon classes with a 90% accuracy on the validation set. This model uses the pre-trained weights from CNN trained with the lung dataset and fine-tune the model by training the output layer using 300 colon images. Furthermore, 3000 colon cell images are dedicated for the validation set. There is some variance over different runs in the data and the average validation accuracy received is 89%.

Transfer learning over the lung and colon cancer histopathological dataset seemed to be an effective way to generate models. Table 15 shows the validation accuracies for predictions using both the colon and lung datasets. The average prediction accuracy of the colon dataset is higher than that of the lung dataset. There might be a few reasons for this. For the lung prediction, the model was pre-trained on the colon dataset which only included 10,000 images and had only two classes. This model was to then predict a dataset that had three classes. In the

Table 15. Results from Pre-Trained CNN

Model	Data Augmentations	Training Dataset	Validation Accuracy
Transfer Learning Using Cell Images	Colon to lung	300	86%
Transfer Learning Using Cell Images	Lung to colon	300	91%

colon prediction case, the model was trained on 15,000 lung images that included three classes. Since the colon dataset was binary and the model was trained on a multi-class problem with more data the validation accuracy was higher when predicting colon images. Figure 10a and Figure 10b show the ROC curves for colon and lung datasets, respectively. Here we assumed disease as the positive class. The ROC graph shows that this model is less effective that the one in Figure 9, used to classify lung images using the proposed method. It can also be seen that the lung adenocarcinoma was similar to the proposed method but there was a huge improve in the ROC when looking at the lung squamous cell carcinoma classification.

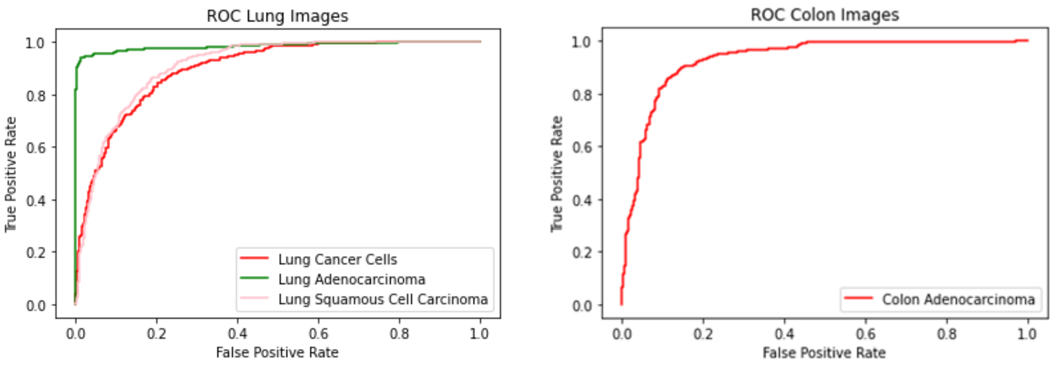

Fig. 10. (10a left) ROC curve of lung classes using a pre-trained CNN, (10b right) ROC curve of colon classes using a pre-trained CNN.

7 CONCLUSIONS

This thesis discussed and implemented various methods for the automatic classification of lung cancer images using a small training set. We were presented with a small lung and colon image dataset consisting of 250 images per class or 1250 images in total. There are three lung classes and two colon classes. We first demonstrated the effect of a small training dataset using a shallow CNN. The maximum accuracy achieved with 750 training data points was only 82% for the lung dataset. Different size training datasets were compared to understand the effect of the number of training samples. Using the VGG16 pre-trained network and various output layers such as ANN, KNN, SVM, and RF, we were able to apply the well-known method of transfer learning. Firstly, we used the augmented dataset which included 12000 images. Transfer learning was applied to different training set sizes with the same classifiers stated previously yielding a highest accuracy of 97.33% with 12000 training datapoints. Secondly, we implemented simple data augmentation techniques (flips, rotations, blur, shear, etc.) on a smaller dataset and utilized this augmented dataset in order to train the final layer of VGG16 network. These techniques yielded a result of 94.6% accuracy while using only a limited number of training samples, i.e. 1200 data points. Next, we performed a more complex data augmentation technique that utilized GANs. These GANs were used to generate a larger dataset. Two implementations of GANs were used: one was trained with only 100 images per class and the other was trained with 250 images per class. These generated images were used with the VGG16 model with multiple classifiers. The ANN network classifier with the VGG16 model achieved a 93.3%.

We observed that it was critical to use enough data-points to train GANs in order to generate reliable new images. However, for most of the scenarios, the original training dataset may not have enough samples to successfully train the GANs. In order to resolve this problem, we proposed a two-level data augmentation technique in which the simple data augmentation technique is followed by a GAN in order to generate new images in a sequential fashion. This proposed two-level data augmentation technique had significantly better validation accuracies from those of one-level data augmentation. With using only 150 original images the proposed two-level data augmentation technique yielded a 94.11% validation accuracy. Lastly, we explored the transfer learning among datasets that are similar in nature – in our case all images were collected from cells. We have replaced the VGG16 network with a shallow CNN pretrained using a colon cancer dataset and tested the network on lung dataset. The vice-versa was also implemented. When trained over the colon dataset to predict lung cancer cells the validation accuracy was 86% and 91% vice versa.

The proposed two-level data augmentation technique yielded a high validation accuracy, 94.11%, with the smallest set of training data (150 samples from the original dataset). The highest validation accuracy recorded was 97.33% obtained with 12000 training images from the original augmented dataset. Note that using two-level data augmentation we can achieve almost the same performance even when the number of training samples is reduced by 98.75% ((12000-150)/12000*100).

REFERENCES

[1] R. L. Siegel, K. D. Miller, and A. Jemal, "Cancer statistics, 2020," *Cancer. J. Clin.,* vol. 70, no. 1, pp. 7-30, 2020.

[2] D. E. Williams, P. C. Pairolero, C. S. Davis, P. E. Bernatz, W. S. Payne, W. F. Taylor, M. A. Uhlenhopp, and R. S. Fontana, "Survival of Patients Surgically Treated for Stage I Lung Cancer," *J. Thoracic Cardio Surgery*, vol. 82, no. 1, pp. 70-76, Dec. 2019. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0022522319393894 [Accessed: 07-Nov-2021].

[3] J. A. Cruz, and D. S. Wishart, "Applications of Machine Learning in Cancer Prediction and Prognosis," *Cancer. Inform.,* vol. 2, pp. 59-77, 2006.

[4] R. Miotto, F. Wang, S. Wang, X. Jiang, and J. T. Dudley, "Deep Learning for Healthcare: Review, Opportunities and Challenges," *Brief. Bioinf.,* vol. 19, no. 6*,* pp. 1236-1246, 2018.

[5] A. Teramoto, T. Tsukamoto, Y. Kiriyama, and H. Fujita, "Automated Classification of Lung Cancer Types from Cytological Images Using Deep Convolutional Neural Networks," *Biomed. Research Int*., vol. 2017, pp. 1-6, 2017.

[6] S. J. Pan, and Q. Yang, "A Survey on Transfer Learning," *IEEE Trans. Knowl. Data Eng.*, vol. 22, no. 10, pp 1345-1359, Oct. 2010.

[7] M. T. Hagos en S. Kant, "Transfer Learning based Detection of Diabetic Retinopathy from Small Dataset," *arXiv [cs.CV]*, May 2019.

[8] Z. J. Wang, R. Turko, O. Shaikh, H. Park, N. Das, F. Hohman, M. Kahng, and D. H. C. Polo, "CNN Explainer: Learning Convolutional Neural Networks with Interactive Visualization," *Tvcg,* vol. 27, no. 2, pp. 1396-1406, 2021.

[9] C. N. Archie, "Chest X-rays Pneumonia Detection using Convolutional Neural Network," Medium, 06-Jun-2020. Accessed: 28-Jun-2021 [Online]. Available: https://towardsdatascience.com/chest-x-rays-pneumonia-detection-using-convolutional-neural-network-63d6ec2d1dee

[10] L. Alzubaidi, J. Zhang, A. J. Humaidi, A. Al-Dujaili, Y. Duan, O. Al-Shamma, J. Santamaría, M. A. Fadhel, M. Al-Amidie, and L. Farhan, "Review of Deep Learning: Concepts, CNN Architectures, Challenges, Applications, Future Directions," *J. Big Data,* vol. 8, no. 1, Mar. 2021.

[11] M. Rubin, O. Stein, N. A. Turko, Y. Nygate, D. Roitshtain, L. Karako, I. Barnea, R. Giryes, and N. T. Shaked, “TOP-GAN: Stain-free cancer cell classification using deep learning with a small training set,” *Med. Image Anal.*, vol. 57, pp. 176-185, 2019.

[12] Karen Simonyan, and Andrew Zisserman, “Very Deep Convolutional Networks for Large-Scale Image Recognition,” *arXiv[cs.CV],* Apr. 2015.

[13] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, A. Rabinovich, “Going Deeper with Convolutions,” *arXiv [cs.CV]*, Sep. 2014.

[14] A. Creswell, T. White, V. Dumoulin, K. Arulkumaran, B. Sengupta, A. A. Bharath, "Generative Adversarial Networks: An Overview," IEEE *Signal Processing Mag*., vol 35, no. 1*,* pp. 53-65, Jan. 2018.

[15] K. Wang C. Gou, Y. Duan, and L. Yilun, "Generative Adversarial Networks:Introduction and Outlook," *IEEE/CAA J. Automatica Sinica,* vol. 4, no. 4, pp. 588-598, Sep. 2017.

[16] D. Berthelot, T. Schumm, and L. Metz, "BEGAN: Boundary Equilibrium Generative Adversarial Networks," *arXiv[cs.LG].* May. 2017.

[17] S. Pascual, A. Bonafonte, and J. Serrà, "SEGAN: Speech Enhancement Generative Adversarial Network," in Proc. *Interspeech,* Aug. 2017, pp. 3642-3646.

[18] S. J. Mobilia, "Classification of hyperspectral colon cancer images using convolutional neural networks," Master's Thesis 5010, Elect. Eng. San José S. Uni. San José. Cal, May 2019.

[19] M. Amrane, S. Oukid, I. Gagaoua, and T. Ensari, "Breast cancer classification using machine learning," in Proc 2018 *IEEE Electric Electron. Comput. Sci. Biomed. Eng. Meeting (EBBT)* Apr. 2018, pp. 1-4.

[20] A. Claudio Quiros, R. Murray-Smith, and K. Yuan, "PathologyGAN: Learning deep representations of cancer tissue," *J. Mach. Learn. Biomed. Imag.,* vol. 2021, pp. 4, 2021.

[21] M. Loey, M. Naman, en H. Zayed, “Deep Transfer Learning in Diagnosing Leukemia in Blood Cells,” *Comput.*, vol. 9, no. 2, 2020.

[22] A. A. Borkowski, M. M. Bui, L. B. Thomas, C. P. Wilson, L. A. DeLand, and S. M. Mastorides, "Lung and Colon Cancer Histopathological Image Dataset (LC25000)," 2019. Distributed by Academic Torrents.

https://academictorrents.com/details/7a638ed187a6180fd6e464b3666a6ea0499af4af

[23] A. A. Borkowski C. P. Wilson, S. A. Borkowski, L. B. Thomas, L. A. Deland, S. J. Grewe, and S. M. Mastorides, “Comparing Artificial Intelligence Platforms for Histopathologic Cancer Diagnosis,” *Fed. Pract. Health Care Prof. VA, DoD, PHS*, vol 36, pp. 456–463, Oct. 2019.

[24] M. D. Bloice, C. Stocker, and A. Holzinger, "Augmentor: An Image Augmentation Library for Machine Learning," *arXiv[cs.CV].* Aug. 2017.

[25] Zou, J., Y. Han, S. S. So, *Overview of Artificial Neural Networks,* Humana Press, 2008.

[26] Z. Zhang, "Introduction to machine learning: k-nearest neighbors," *Ann. Translational Med.,* vol. 4, no. 11, pp. 218, Jun. 2016.

[27] C. L. M. Morais, and K. M. G. Lima, "Comparing unfolded and two-dimensional discriminant analysis and support vector machines for classification of EEM data," *Chemometrics Intell. Lab. Syst.,* vol. 170, pp. 1-12, Sep. 2017.

[28] A. Géron, Hands-on Machine Learning with Scikit-Learn and TensorFlow : Concepts, Tools, and Techniques to Build Intelligent Systems, Sebastopol, CA, USA: O’Reilly Media, 2017.

[29] M. F. Safdar, S. Al-Kobaisi, F. T. Zahra, “A comparative analysis of data augmentation approaches for magnetic resonance imaging (MRI) scan images of brain tumor,” *Acta Inform. Medica*, vol 28, pp. 29-36, Feb. 2020.

[30] A. Radford, L. Metz, and S. Chintala, “Unsupervised representation learning with deep convolutional generative adversarial networks,” in Proc. *Int. Conf. Learn. Representations (ICLR)*, 2016, pp 1-16.

[31] D. P. Kingma, and J. Ba, "Adam: A Method for Stochastic Optimization," *arXiv[cs.LG]* Dec. 2014.