**Deep Learning and Neural Network Project Proposal**

Project Title: Lung and Colon Cancer Detection

**Authors:**

Mehta, Unnati Sachin, [mehta.un@northeastern.edu](mailto:mehta.un@northeastern.edu)

Chattoraj, Ankuran, [chattoraj.a@northeastern.edu](mailto:chattoraj.a@northeastern.edu)

Singh, Akarsh, [singh.akar@northeastern.edu](mailto:singh.akar@northeastern.edu)

1. **Abstract**

Lung and colon cancer pose significant health challenges, necessitating early detection for successful treatment. This research introduces an innovative strategy for cancer detection utilizing Neural Networks. In our study, we propose a new method for detecting lung and colon cancer, emphasizing the crucial role of early detection in effective treatment. Our approach revolves around various Neural Networks rather than specifically CNNs. Our method demonstrated a notable level of accuracy in detecting these cancers. The methodology involves training a neural network model on a comprehensive dataset of lung and colon cancer images and subsequently employing this model to predict cancer presence in new images. Through rigorous evaluation on a test set, our approach achieved a high accuracy level of 98%. These results underscore the potential of Neural Network-based techniques in enhancing the precision and efficiency of cancer diagnosis, representing a significant advancement in improving healthcare outcomes for cancer detection and treatment.

1. **Introduction**

The World Health Organization reports that cancer is the leading global cause of death, and estimates suggest a 47% increase in global cancer cases to 28.4 million by 2040 [1]. Among cancers, lung and colorectal (colon and rectum) cancers rank high after breast cancer, with incidence rates of 11.4% and 10%, respectively, in 2020 [2]. Despite being relatively low, there is a possibility of synchronous occurrence of lung and colon cancers [3]. Furthermore, these two cancers have the highest mortality rates at 18% for lung cancer and 9.4% for colorectal cancer among all cancer types. These cancers are caused by a spectrum of factors, including genetic and environmental influences, and can lead to severe health complications if not diagnosed and treated promptly.

The detection and timely treatment of Lung and Colon Cancer are crucial due to the severe health complications associated with these diseases. Traditional diagnostic methods face challenges in early identification. In recent years, artificial intelligence (AI) and machine learning (ML), particularly Neural Networks, have gained interest for their potential in aiding cancer detection [4]. Neural Networks, adept at analyzing medical images, show promise in swiftly and precisely identifying cancerous growths in the lungs and colon. Similar to their success in pneumonia detection, Convolutional Neural Networks (CNNs) can assist healthcare professionals by analyzing medical images for indications of cancer.

However, applying Neural Networks for cancer detection presents challenges, including the need for comprehensive training data and robust evaluation metrics. The ambiguous symptoms of these cancers emphasize the importance of advanced computational methods [5]. Computer-Aided Design (CAD) tools, integral in AI and ML research, have demonstrated potential in aiding medical professionals by extracting crucial features from medical images for disease detection and classification, including breast and lung cancer.

Despite challenges, the evolving landscape of Neural Networks in cancer detection signals promising advancements in the accurate diagnosis and treatment of Lung and Colon Cancer. The integration of sophisticated computational methods in medical imaging holds the potential to significantly enhance the capabilities of healthcare professionals in addressing these challenging diseases.

1. **Background**

Various studies emerged over the years that utilized contemporary AI methods for lung and colon cancer detection. The following studies were based on the LC25000 dataset used for this project [6]. Authors of [7] introduced a hybrid ensemble model for identifying lung and colon cancer, achieving accuracy rates of 99.05% for lung cancer, 100% for colon cancer, and 99.30% for combined detection. Results suggested the model significantly outperformed existing ones, indicating potential clinical applicability for assisting doctors in cancer diagnosis. [8] The study developed three strategies for early diagnosis using histological images from the LC25000 dataset. GoogLeNet and VGG-19 models were employed, and redundant features were reduced through PCA. The third strategy, combining CNN models and handcrafted features, achieved impressive results: 99.85% sensitivity, 100% precision, 99.64% accuracy, 100% specificity, and a 99.86% AUC. In the next study [9], inefficient features were identified using Equilibrium and Manta Ray Foraging algorithms from the DarkNet-19 model. These were separated to form an efficient feature set (complementary rule insets), combined, and classified using Support Vector Machine (SVM). The classification process achieved an impressive 99.69% accuracy, showcasing improved performance with the complementary method and optimization algorithms. Masud et al. [10] introduced a classification framework using Deep Learning and Digital Image Processing to distinguish between five types of lung and colon tissues. The proposed model achieved a maximum accuracy of 96.33% in identifying cancer tissues, offering a potential automated and reliable system for medical professionals in detecting various types of lung and colon cancers. Finally, a pretrained neural network (AlexNet) was tuned by Mehmood et al. and co-authors [11]. They modified four layers, achieving an initial accuracy of 89%. To enhance accuracy without compromising efficiency, a contrast enhancement technique was applied to the underperforming class. This improved overall accuracy to 98.4%, demonstrating improved performance and computational efficiency.

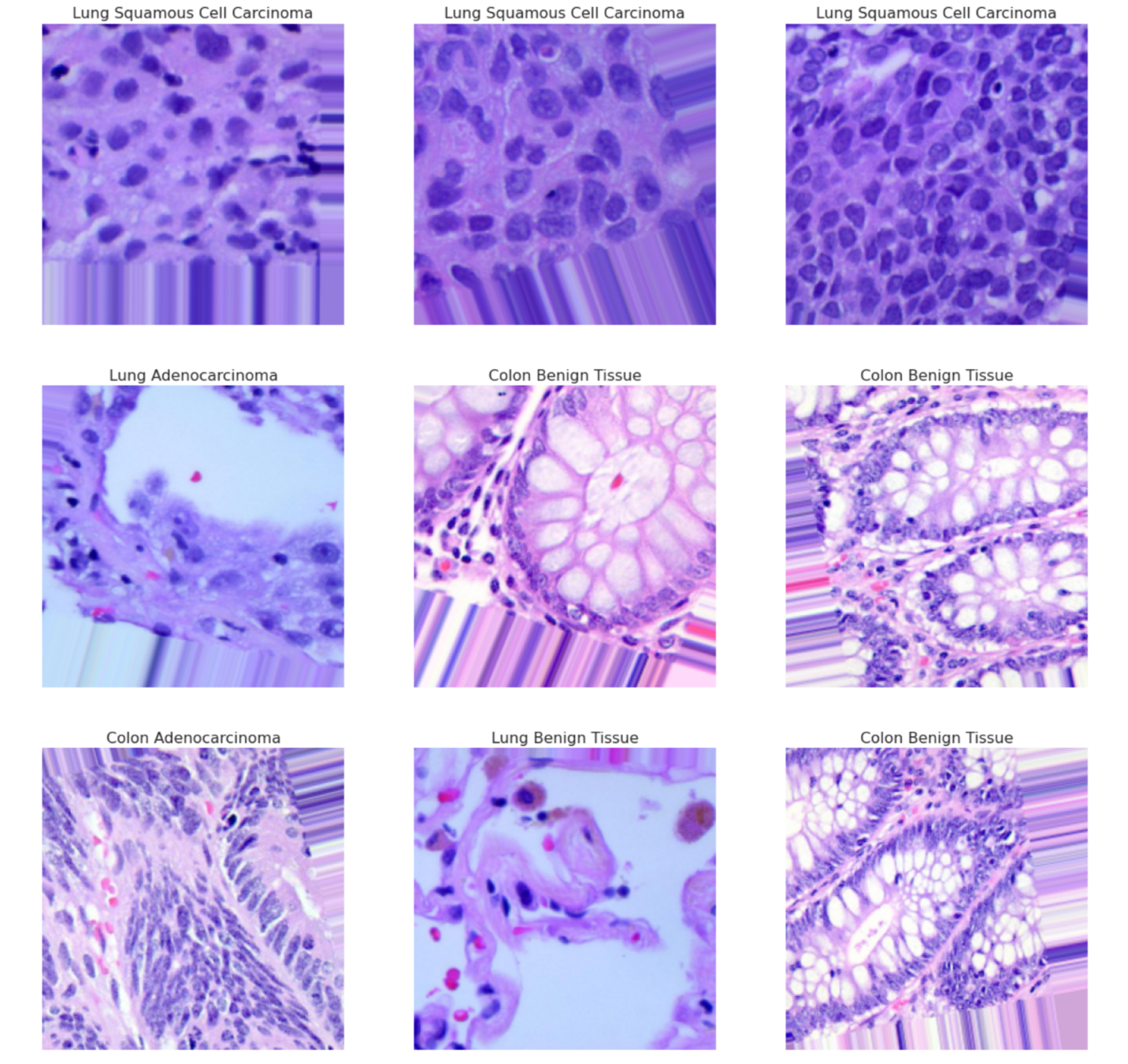
Referring to these highly-cited, accomplished studies. The dataset sets a high bar for histological image processing tasks. We aspire to make the most of our course material to achieve good results with respect to this dataset.

1. **Approach**
   1. **Dataset Description**

Link to the dataset: Lung\_Colon\_Dataset

The dataset contains color 25,000 images with 5 classes of 5,000 images each. All images are 768 x 768 pixels in size and are in jpeg file format. Our dataset can be downloaded as a 1.85 GB zip file LC25000.zip. After unzipping, the main folder lung\_colon\_image\_set contains two subfolders: colon\_image\_sets and lung\_image\_sets. The subfolder colon\_image\_sets contains two secondary subfolders: colon\_aca subfolder with 5000 images of colon adenocarcinomas and colon\_n subfolder with 5000 images of benign colonic tissues. The subfolder lung\_image\_sets contains three secondary subfolders: lung\_aca subfolder with 5000 images of lung adenocarcinomas, lung\_scc subfolder with 5000 images of lung squamous cell carcinomas, and lung\_n subfolder with 5000 images of benign lung tissues.

Figure 1: Overview of Samples from all 5 Classes



Given that the dataset was fairly equally distributed and the image size had consistent dimensions as stated above, we did not have to bother with image generation or up-sampling to maintain class balance. We used TensorFlow Keras library for training our models.

* 1. **Data Loading Configuration**

We employed the TensorFlow Keras library to implement an ImageDataGenerator for our classification task. The images, sourced from dataframes (that saved image paths), were preprocessed and augmented to enhance the model's generalization capabilities during training. The training data underwent transformations such as rotation, width and height shifts, shear, zoom, and horizontal/vertical flips, contributing to a more diverse dataset. The generator produced batches of images and corresponding labels for both training and validation sets, facilitating efficient and scalable model training. Additionally, the normalization step ensured pixel values were within the [0, 1] range. This comprehensive data preparation pipeline laid the foundation for training convolutional neural networks to achieve robust performance on image classification tasks.

* 1. **Iterative CNN Design**

We started off with a baseline CNN Model and based on performance iteratively improved model depth. After fitting versions of custom CNNs we adapted transfer learning techniques to see how well pre-trained models could fit. Below are the layers that were used throughout the study.

**Convolutional Layer:** In a Convolutional Neural Network (CNN), the convolutional layer is responsible for processing input images through the application of convolutional operations. The input image is represented as a matrix, and a convolution operation is performed with a feature detector or filter. This filter, often of size 3x3, scans the input matrix, and the result is a feature map. These convolutional operations help the network identify patterns, edges, and features within the image.

**Pooling Layer:** Following the convolutional layer, a pooling layer is used to down-sample the feature maps, reducing their spatial dimensions. Pooling is typically done using operations like max pooling, where the maximum value in a certain region is retained, or average pooling, where the average value is computed. The pooling layer helps in reducing the computational load and focusing on the most relevant information, making the network more robust and efficient.

**Fully Connected Layers:** After the convolutional and pooling layers, the feature maps are flattened into a column vector. This vector is then fed into one or more fully connected layers, forming the traditional neural network structure. These fully connected layers enable the network to learn complex relationships and make predictions based on the hierarchical features extracted from the input image.

**Batch Normalization:** Batch Normalization (BatchNorm) is a technique in deep learning that helps stabilize and accelerate the training of neural networks. During training, BatchNorm normalizes the input of each layer by adjusting and scaling the activations. This helps combat issues like internal covariate shift, where the distribution of inputs to a layer changes over time, hindering convergence. By normalizing the input within each mini-batch, BatchNorm allows for more stable and faster training. Moreover, it acts as a regularizer, reducing the dependence on dropout and allowing for higher learning rates

**Dropout:** Dropout is a regularization technique used to prevent overfitting in neural networks. During training, Dropout randomly sets a fraction of input units to zero at each update, effectively "dropping out" these units. This introduces redundancy and prevents the model from relying too heavily on specific neurons, making it more robust and reducing the risk of overfitting to the training data. Dropout acts as a form of ensemble learning, as it trains different versions of the model with random subsets of neurons dropped out at each iteration.

In summary, a CNN is designed to automatically and adaptively learn spatial hierarchies of features from images. The convolutional and pooling layers capture local patterns and reduce spatial dimensions, the fully connected layers analyze global relationships and make predictions. While, dropout and normalization helps speed up training and prevents overfitting. This architecture has proven highly effective in image recognition, object detection, and other computer vision tasks.

1. **Results**
   1. ***Model Architecture (Baseline Custom CNN):***

***Input Layer***: Accepts input data with shape (80, 80, 3), indicating an image size of 80x80 pixels with three color channels (RGB).

***Convolutional Layer 1:***

Number of filters: 32 Filter size: 3x3; Activation function: Rectified Linear Unit (ReLU); Followed by MaxPooling with pool size 2x2

***Convolutional Layer 2***:

Number of filters: 64; Filter size: 3x3; Activation function: ReLU; Followed by MaxPooling with pool size 2x2 and dropout regularization of 20%

***Convolutional Layer 3:***

Number of filters: 128; Filter size: 3x3;Activation function: ReLU; Padding: 'same' (to maintain the spatial dimensions); Followed by MaxPooling with pool size 2x2 and dropout regularization of 20%

***Flattening Layer:*** Flattens the output from the convolutional layers to prepare for the fully connected layer.

***Fully Connected Layer:***

Number of neurons: 256; Activation function: ReLU

***Output Layer:***

Number of neurons: Equal to the number of output classes (specified as classes); Activation function: Softmax (for multi-class classification)

***Model Compilation:***

The model is compiled using the Adam optimizer, categorical cross-entropy as the loss function (suitable for multi-class classification), and accuracy as the evaluation metric.

Training:

The model is trained using the fit method:

· batch\_size=64, epochs=25

* 1. ***Model Architecture (Deeper CNN with BatchNorm):***

***Input Layer***: Accepts input data with shape (224, 224, 3), indicating an image size of 80x80 pixels with three color channels (RGB).

***Convolutional Block 1:***

2x (Number of filters: 64 Filter size: 3x3; Activation function: Rectified Linear Unit (ReLU); Padding ‘same’) Followed by MaxPooling with pool size 2x2; BatchNorm; Dropout: 0.2

***Convolutional Block 2***:

2x (Number of filters: 128 Filter size: 3x3; Activation function: Rectified Linear Unit (ReLU); Padding ‘same’) Followed by MaxPooling with pool size 2x2; BatchNorm; Dropout: 0.2

***Convolutional Block 3:***

Number of filters: 256; Filter size: 3x3;Activation function: ReLU; Padding: 'same' (to maintain the spatial dimensions); Followed by MaxPooling with pool size 2x2; BatchNorm; Dropout: 0.2

***Convolutional Block 4:***

Number of filters: 512; Filter size: 3x3;Activation function: ReLU; Padding: 'same' (to maintain the spatial dimensions); Followed by MaxPooling with pool size 2x2; BatchNorm; Dropout: 0.2

***Convolutional Block 5:***

Number of filters: 512; Filter size: 3x3;Activation function: ReLU; Padding: 'same' (to maintain the spatial dimensions); Followed by MaxPooling with pool size 2x2; BatchNorm; Dropout: 0.2

***Flattening Layer:*** Flattens the output from the convolutional layers to prepare for the fully connected layer.

***Flattening Layer:*** Flattens the output from the convolutional layers to prepare for the fully connected layer.

***Fully Connected Layer 1:***

Number of neurons: 256; Activation function: ReLU; BatchNorm; Dropout: 0.2

***Fully Connected Layer 2:***

Number of neurons: 256; Activation function: ReLU; BatchNorm; Dropout: 0.2

***Output Layer:***

Number of neurons: Equal to the number of output classes (specified as classes); Activation function: Softmax (for multi-class classification)

***Model Compilation:***

The model is compiled using the Adam optimizer, categorical cross-entropy as the loss function (suitable for multi-class classification), and accuracy as the evaluation metric.

Training:

The model is trained using the fit method:

· batch\_size=64, epochs=25

* 1. **ResNet50 Model Architecture**

***Residual Blocks:***

The model includes two types of blocks:

*1. Identity Block (identity\_block):*

This block contains three convolutional layers with skip connections (shortcuts) that bypass these layers to help the model learn the residual (difference) between the input and output.

The activation function used throughout is ReLU (Rectified Linear Unit).

2. Convolutional Block (convolutional\_block):

Similar to the identity block but includes a convolutional layer in the shortcut path when the input and output dimensions don't match due to strides.

***Input Layer:***

Accepts input images of shape (224, 224, 3)

***Pre-processing:***

Applies zero-padding to the input images.

***Initial Convolutional Layer:***

Performs a convolution operation with 64 filters of size 7x7 and a stride of 2, followed by batch normalization, ReLU activation, and max-pooling to downsample the feature maps.

***Residual Blocks:***

Includes multiple instances of convolutional and identity blocks:

These blocks have different numbers of filters, namely [64, 64, 256], [128, 128, 512], [256, 256, 1024], and [512, 512, 2048], reflecting the increasing complexity and depth of the network.

***Global Average Pooling:***

Reduces the spatial dimensions of the feature maps to a vector by computing the average of each feature map.

***Fully Connected Layer:***

A dense layer with the number of neurons equal to the specified number of classes (5 in this case) and a softmax activation function for multi-class classification.

***Model Compilation:***

The model is compiled using the Adam optimizer, categorical cross-entropy as the loss function (suitable for multi-class classification), and accuracy as the evaluation metric.

***Training:***

The model is trained using the fit method:

· batch\_size=64, epochs=10

* 1. **ResNet101 Model Architecture**

***Base ResNet101 Model:***

An instance of the ResNet101 model is added to a Sequential model (resnet). The ResNet101 is imported with weights pretrained on ImageNet and set to exclude the top classification layers (include\_top=False). nput\_shape=Image\_Size+[3]: Specifies the input shape for the model based on the given Image\_Size (224x224) and three color channels (RGB). Pooling='average': Uses global average pooling to reduce the spatial dimensions and generate a pooled feature representation.

***Freezing Layers:***

The layers won't be updated during subsequent training, preserving the learned features from ImageNet.

***Flatten Layer:***

Flattens the output from the pre-trained ResNet101 base model to prepare for the additional dense layers.

***Dense Layers:***

Dense(256, activation='relu'): Contains 256 neurons with ReLU activation.

Dense(128, activation='relu'): Contains 128 neurons with ReLU activation.

***Dropout Layer:***

· Introduces a dropout layer with a dropout rate of 20% after the second dense layer to prevent overfitting by randomly dropping 20% of the neurons during training.

***Output Layer:***

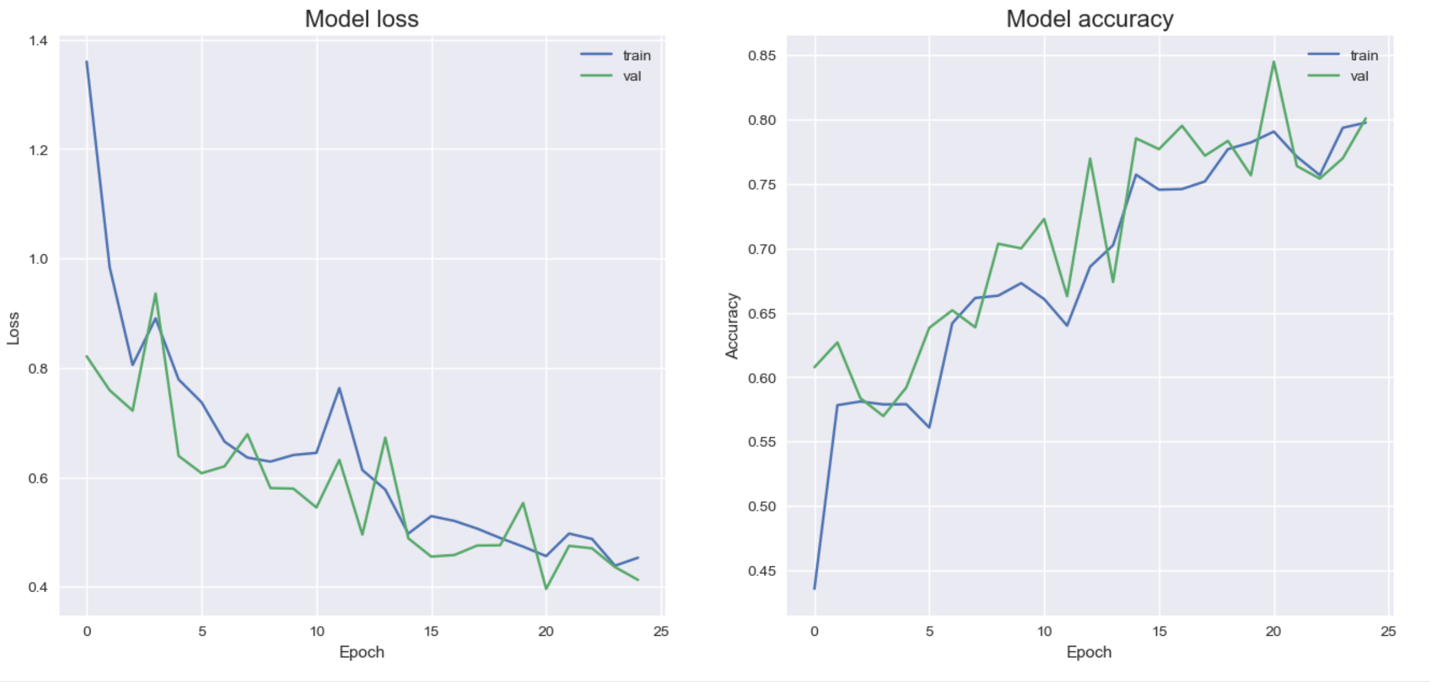
A dense layer with 5 neurons and a softmax activation function for multi-class classification. This layer generates predictions for the five specified classes.

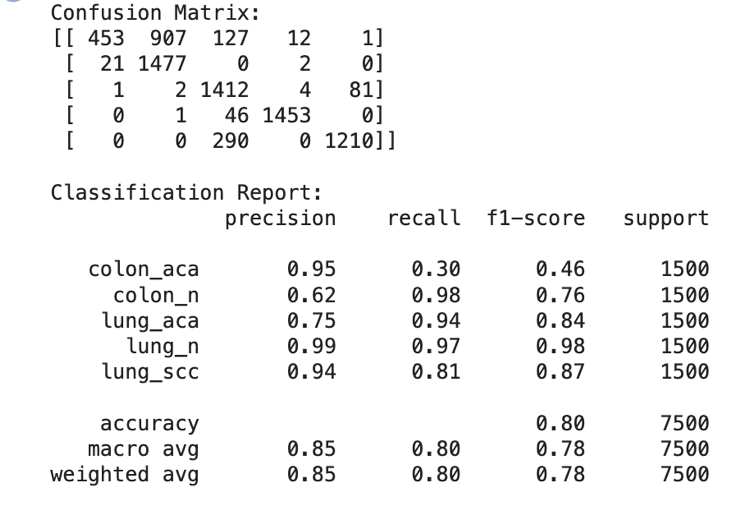
***Model Compilation:***

The model is compiled using the Adam optimizer, categorical cross-entropy as the loss function (suitable for multi-class classification), and accuracy as the evaluation metric; epochs=5.

* 1. ***Performance Results***

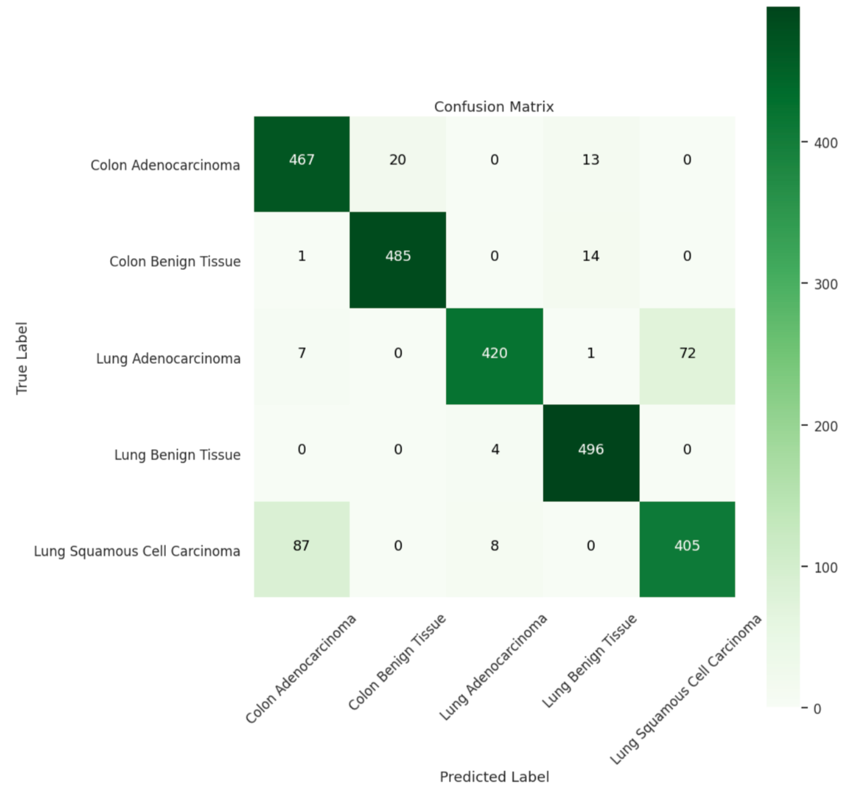
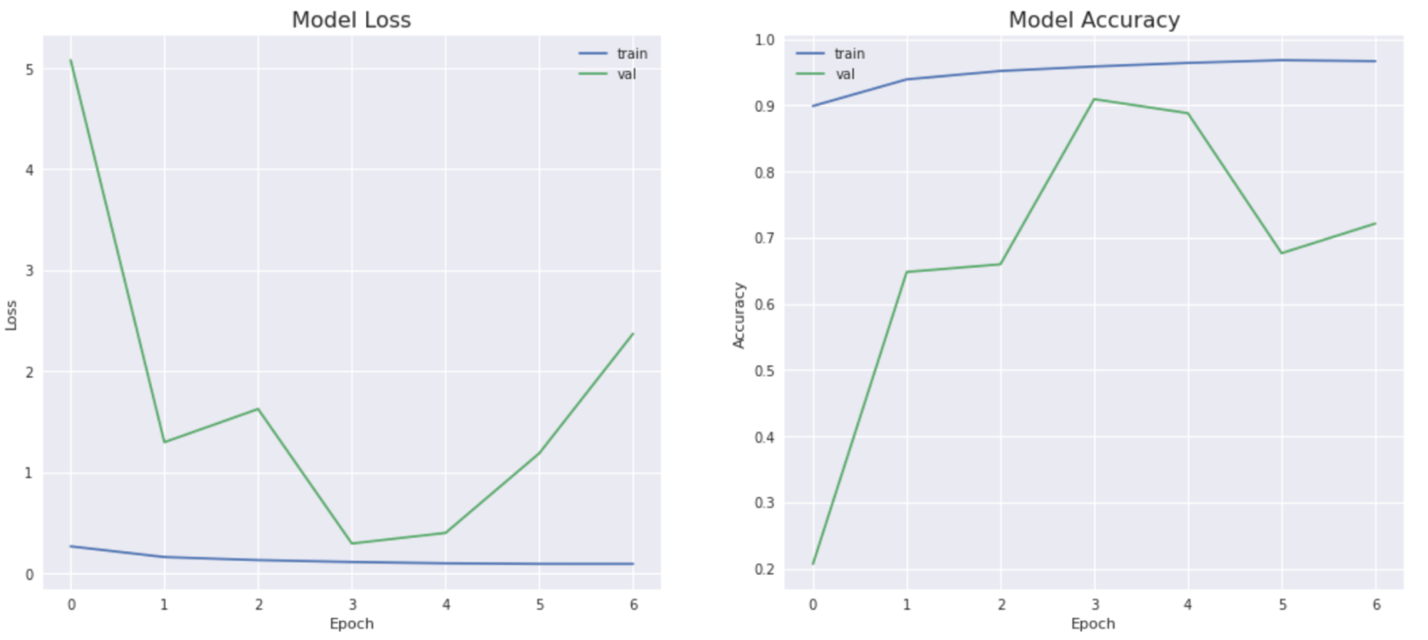
Figure 2: Baseline CNN Model Performance



****

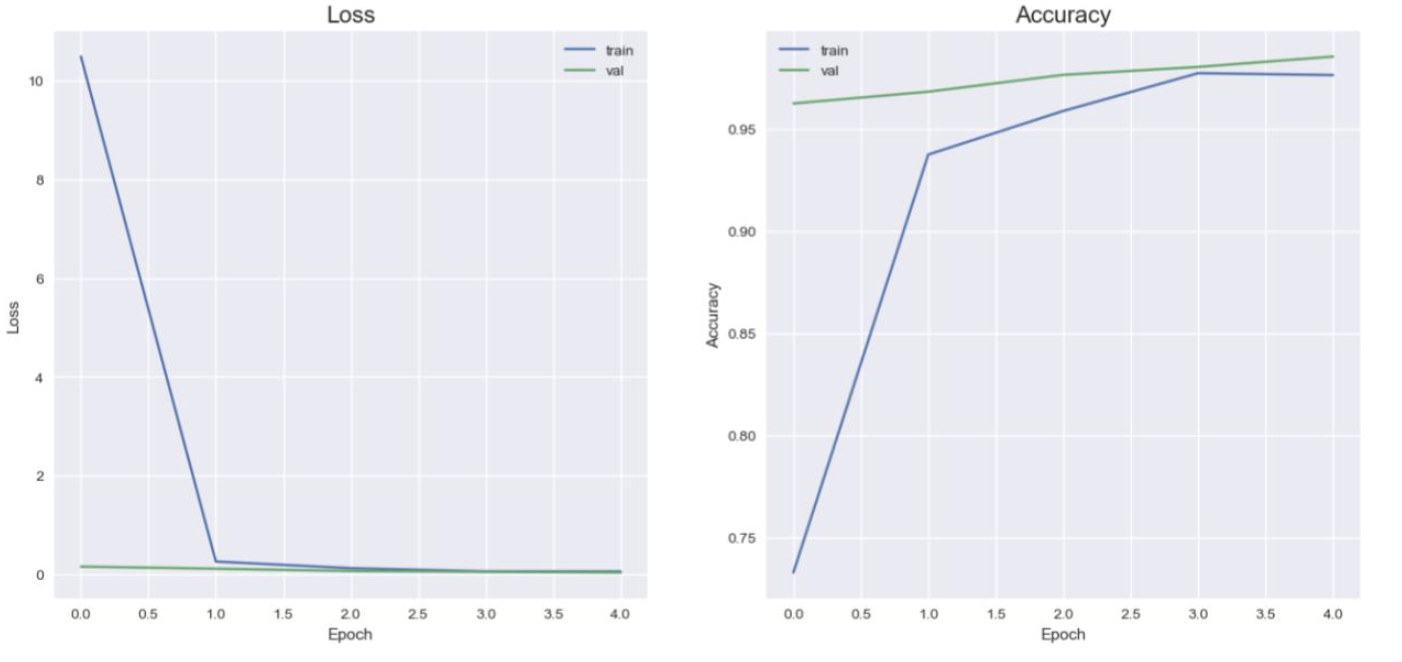
Our baseline model results were comparatively accurate given the least amount of trainable parameters. The confusion matrix and classification report provide a comprehensive evaluation of the model's performance on a multi-class classification task (Figure 2). For instance, the model performed well in identifying 'colon\_n' and 'lung\_n' classes with high precision and recall, while facing challenges in accurately classifying 'colon\_aca.' The classification report further details precision, recall, and F1-score for each class, along with macro and weighted averages. The overall accuracy of 80% indicates a reasonably good performance, with notable strengths in certain classes

Figure 3: Deeper CNN with BatchNorm Model Performance

****

Despite higher computational cost this model showed higher predictive power across all categories in most metrics shown in Figure 3. The best performance was achieved with non-cancerous lung tissue. Early stopping was used to prevent over-fitting and the best performance was achieved on epoch 4 hence, epoch 4 parameter were used for inference, results in a 92% accuracy on the test set.

Figure 4: Deeper CNN with BatchNorm Model Performance



**Discussion**

The CNN model seems to be underperforming with relatively low accuracies across all sets. The sudden drop in loss at epoch 1 might indicate that the model starts learning initially, but it doesn't improve significantly beyond that point. The model might be too simplistic or might lack the depth required to learn complex patterns in the data.

The ResNet50 model shows improvement in training accuracy compared to the CNN model. However, the validation and test accuracies are quite low. The sudden drop in loss at epoch 1 might indicate a learning burst, but the model might not be able to generalize well to unseen data, suggesting possible overfitting due to the large difference between training and validation accuracies.

The ResNet101 model demonstrates significantly higher performance compared to the other models. It achieves high accuracies on both the validation and test sets, indicating good generalization and learning capabilities. The sudden drop in loss at epoch 1 followed by high accuracies on both validation and test sets suggests that the model is learning effectively and generalizing well.

CNN performs poorly across all metrics, indicating that it might be too simple for the task or not trained sufficiently. ResNet50 performs better than CNN in training but shows signs of overfitting as seen from the low validation and test accuracies. ResNet101 outperforms both the CNN and ResNet50 models significantly. It exhibits high accuracies on the validation and test sets, suggesting that it effectively learns the features of the data and generalizes well to unseen samples.

**References**

1. de Martel, C.; Georges, D.; Bray, F.; Ferlay, J.; Clifford, G.M. Global Burden of Cancer Attributable to Infections in 2018: A Worldwide Incidence Analysis. *Lancet. Glob. Health* **2020**, *8*, e180–e190.
2. Sung, H.; Ferlay, J.; Siegel, R.L.; Laversanne, M.; Soerjomataram, I.; Jemal, A.; Bray, F. Global Cancer Statistics 2020: GLOBOCAN Estimates of Incidence and Mortality Worldwide for 36 Cancers in 185 Countries. CA. *Cancer J. Clin.* **2021**, *71*, 209–249.
3. Kurishima, K.; Miyazaki, K.; Watanabe, H.; Shiozawa, T.; Ishikawa, H.; Satoh, H.; Hizawa, N. Lung Cancer Patients with Synchronous Colon Cancer. *Mol. Clin. Oncol.* **2018**, *8*, 137–140.
4. Ladbury C, Amini A, Govindarajan A, Mambetsariev I, Raz DJ, Massarelli E, Williams T, Rodin A, Salgia R. Integration of artificial intelligence in lung cancer: Rise of the machine. Cell Rep Med. 2023 Feb 21;4(2):100933. doi: 10.1016/j.xcrm.2023.100933. Epub 2023 Feb 3. PMID: 36738739; PMCID: PMC9975283.
5. Ahn JS, Shin S, Yang SA, Park EK, Kim KH, Cho SI, Ock CY, Kim S. Artificial Intelligence in Breast Cancer Diagnosis and Personalized Medicine. J Breast Cancer. 2023 Oct;26(5):405-435. doi: 10.4048/jbc.2023.26.e45. PMID: 37926067; PMCID: PMC10625863.
6. Borkowski, A. A., Bui, M. M., Thomas, L. B., Wilson, C. P., DeLand, L. A., & Mastorides, S. M. (2019). Lung and colon cancer histopathological image dataset (lc25000). *arXiv preprint arXiv:1912.12142*.
7. Talukder, M. A., Islam, M. M., Uddin, M. A., Akhter, A., Hasan, K. F., & Moni, M. A. (2022). Machine learning-based lung and colon cancer detection using deep feature extraction and ensemble learning. *Expert Systems with Applications*, *205*, 117695.
8. Al-Jabbar M, Alshahrani M, Senan EM, Ahmed IA. Histopathological Analysis for Detecting Lung and Colon Cancer Malignancies Using Hybrid Systems with Fused Features. Bioengineering (Basel). 2023 Mar 21;10(3):383. doi: 10.3390/bioengineering10030383. PMID: 36978774; PMCID: PMC10045080.
9. Toğaçar M. Disease type detection in lung and colon cancer images using the complement approach of inefficient sets. Comput Biol Med. 2021 Oct;137:104827. doi: 10.1016/j.compbiomed.2021.104827. Epub 2021 Sep 2. PMID: 34560401.
10. Masud M, Sikder N, Nahid AA, Bairagi AK, AlZain MA. A Machine Learning Approach to Diagnosing Lung and Colon Cancer Using a Deep Learning-Based Classification Framework. Sensors (Basel). 2021 Jan 22;21(3):748. doi: 10.3390/s21030748. PMID: 33499364; PMCID: PMC7865416.
11. Mehmood, S., Ghazal, T. M., Khan, M. A., Zubair, M., Naseem, M. T., Faiz, T., & Ahmad, M. (2022). Malignancy detection in lung and colon histopathology images using transfer learning with class selective image processing. *IEEE Access*, *10*, 25657-25668