DISEASE DIAGNOSIS FROM MEDICAL IMAGING

DONE BY

TATIREDDY NANDEESWAR REDDY	**22101A010339**
M.SURYA PRAKASH	**22101A010487**
SEGU ARAVINDA SREE	**22101A010332**

OF CLASS AIML B5
B.TECH 4TH SEMESTER
2023-2024

Under the Guidance of

Dr. J. Avanija
Professor
Dept. of Aiml
School of Computing
Mohan Babu University

Dr. B. Narendra Kumar Rao
Prof & head
Dept. of AIML
School of Computing
Mohan Babu University

MOHAN BABU UNIVERSITY
Sree Sainath Nagar, Tirupati - 517 102

Tatireddy Nandeeswar Reddy : tatireddynandeeswarreddy@gmail.com
(Phone Number : 9885841535)
M. Surya Prakash : suryaprakash5466225@gmail.com
(Phone Number : 9381136965)
Segu Aravinda Sree : seguaravindasree@gmail.com
(Phone Number : 6300713557)

ABSTRACT

Precise and well-timed diagnosis of the most common and fatal types of cancers which are lung and colon cancers plays a crucial role to advance patient outcome and decreasing death rates. With the upcoming display of Machine Learning and Deep Learning techniques, there is an unexpected opportunity to increase the diagnostic accuracy and successfulness of cancer detection from medical imaging.This project gives a complete review of recent improvements in lung and colon cancers diagnosis using ML and DL techniques concerned to medical imaging data. This review starts by defining the present challenges in cancer diagnosis from medical imaging, including difficulties related to variability in image interpretation and the need for more strong and automated analysis methods.Accuracy using SVM, and DL techniques, including 2D convolution neural network, Convolutional auto-encoder neural network and are discussed with reference to their application to lung and colon cancer diagnosis.

Keywords : Disease Diagnosis, Support Vector Machine, Activation Function: Rectified Linear unit (ReLu), 2D Convolutional Neural Networks, Histopathological Images

Table Of Contents

Chapter 1

1.1 INTRODUCTION

Lung and colon cancer are the most common public health problems occurring worldwide, Advanced imaging process such as CT, MRI, and X-Ray are an important parts of finding out the tumors in the human body.The use of the Machine Learning and Deep Learning Concepts in the advanced automation of cancer diagnosis is important for the betterment in the treatment of patients and give proper treatment.

1.2 OBJECTIVES

The main aim of the project is to develop improved machine learning algorithms to diagnose early lung and colon cancer, decreasing overall death rates and improved patient outcomes.The goal is to increase the accuracy of disease diagnosis, decrease false positives and false negatives and provide customized treatment planning. The algorithms should be in cooperation with active clinical workflows, establishing practicality and efficiency for healthcare professionals. Algorithms should be capable of performing better across different datasets ,patient demographics and imaging techniques.

1.3 APPLICATIONS

Machine Learning models can very much improve healthcare by assisting in early diagnosing of lung and colon cancer from medical imaging data. These models can support radiologists in making appropriate decisions about treatment and predicting treatment response based on previous imaging data. This automated approach can improve treatment selection and patient outcomes, eventually directing to better healthcare outcomes.

1.4 LIMITATIONS

- Lack of performance differentiation between the suggested method and standard machine learning methods.
- Deep Learning approaches for the detection of all diseases considering noise removal from the given dataset.
- We can work on classification between various types of cancers with this model.
- We seek to examine the features learned by by 3D CNN with the help of ‘explainable AI’ methods.
- Developing specialized transfer learning techniques.
- We intent to work on architecture of the classification model and engineer new set of features from histopathological images.
- We have to think about how can we come up with methods that depend on less data to train, which can still generate well.

Chapter 2
Literature Survey

1. Two DL techniques and many ML methods for identifying brain tumors using MRI are suggested. The 2D CNN and auto- encoder network were improved by taking 3264 MRI brain images dataset.The 2D CNN reached high accuracy and average recall values, making it correct for physicians and radiologists in clinical systems for brain tumor identification.
2. Usage of DL and ML in medical diagnosis and sorting of many diseases are explored. ML classifiers and DL techniques are compared using MRI datasets and examines 20 studies. The results direct to help healthcare practitioners to select suitable methods with high accuracy and less time.
3. Breast cancer identification accuracy with the help of CAD system, using SVM, KNN, LR methods to separate normal and abnormal growth of cell, decreasing false-negatives and false-positives.
4. A method using image processing along with ML techniques to discover and classify bone cancer, with 85% accuracy is shown in the paper. The given method uses SVM and Random Forest models, with HOG feature sets, and exceeds previous work higher recall and accuracy.
5. Pa-DBN-BC, a patch-based DL method for discovering and classifying breast cancer on histopathological image dataset using Deep Belief Network is given. This method draws out features from image patches along pre-tuning and fine tuning along with logistic regression to classify patches. The model reached 86% accuracy over a slide histopathological image dataset, exceeding previous methods and decreasing computational costs.

Chapter 3
Methodology

3.1 Existing System

The existing system gives the description about a 2D Convolutional Neural Network architecture using 9792 data, which has 90% of training data and 10% of testing data. The existing system has different layers of network which include convolution, with a hierarchical structure and filling nearby cells. The result of convolutional layer is looked based on size of the output of the previous convolution layer, filters, masked layers in fully connected layers. The network contains

of 2D multilayer convolutional networks for encoder and decoder. The aim is to classify MRI images into meningioma, pituitary gland tumor, glioma, health brain tumor classes with metrics like accuracy, recall, F measure, etc. Training accuracy of 2D CNN was 96.4752% and training accuracy of convolutional auto encoder was 95.6371%. The study also differentiates traditional machine learning classifiers like SVM, Logical Regression, Random Forest, Stochastic Gradient Descent, Nearest Neighbor, Multilayer perceptron and the highest accuracy was found for each set of brain images.

3.2 Proposed System

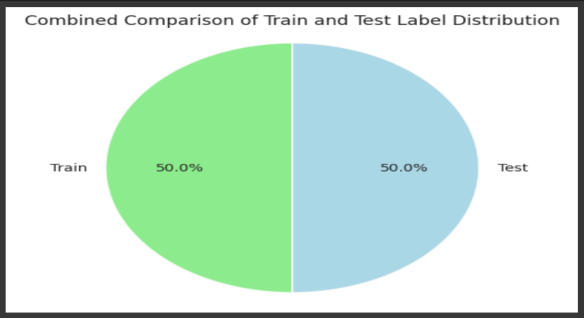
The code is a python for DL utilizing tenser flow and keras to train CNN on image data. For accessing the files imported libraries and modules such as Tensorflow, Keras, OpenCV and mounts GoogleDrive are used. Iterates through subdirectories to collect files paths and corresponding labels into a pandas DataFrame('df'). It splits the data into training and testing sets from sk-learn. It configure ImageDataGenerator for data augmentation and flow from DataFrame for training, validation and testing datasets.Define a sequential Keras model consists of multiple CNN and pooling layers.

Fitting the model on training data and validating it on validation data is how the model is trained. Trains the model for 5 epochs and stores the training history.

(1)Count Plot for Train and Test Set Label Distribution

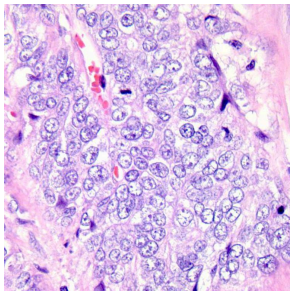
(2)Comparison of train and test label dataset

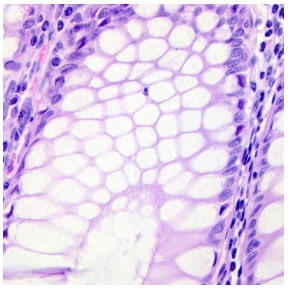
```
import matplotlib.pyplot as plt
labels = ['Train', 'Test']
sizes = [len(y_train), len(y_test)]
colors = ['lightgreen', 'lightblue']

plt.pie(sizes, labels=labels, autopct='%1.1f%%', colors=colors, startangle=90)
plt.axis('equal')
plt.title('Combined Comparison of Train and Test Label Distribution')
plt.show()
```

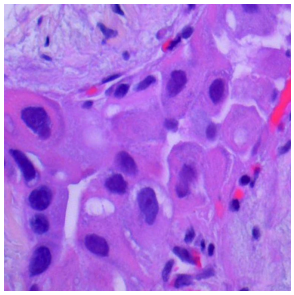


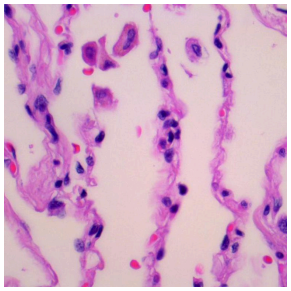
3.2.1 Dataset Description

The dataset used is Lung and Colon Cancer Histopathological Images 2019 data. There are 25000 histopathological images in 5 classes in the dataset. All the images are in jpeg format with a resolution of 768 x 768 pixels.
The dataset contains total 750 images of lung tissue(250 benign lung tissue, 250 lung adenocarcinomas, 250 lung squamous cell carcinomas) and 500 total images of colon tissue(250 benign colon tissue, 250 colon adenocarcinomas) were created from an original sample of HIPAA compliant and validated sources, they were expanded to 25000 using the Augmentor package.

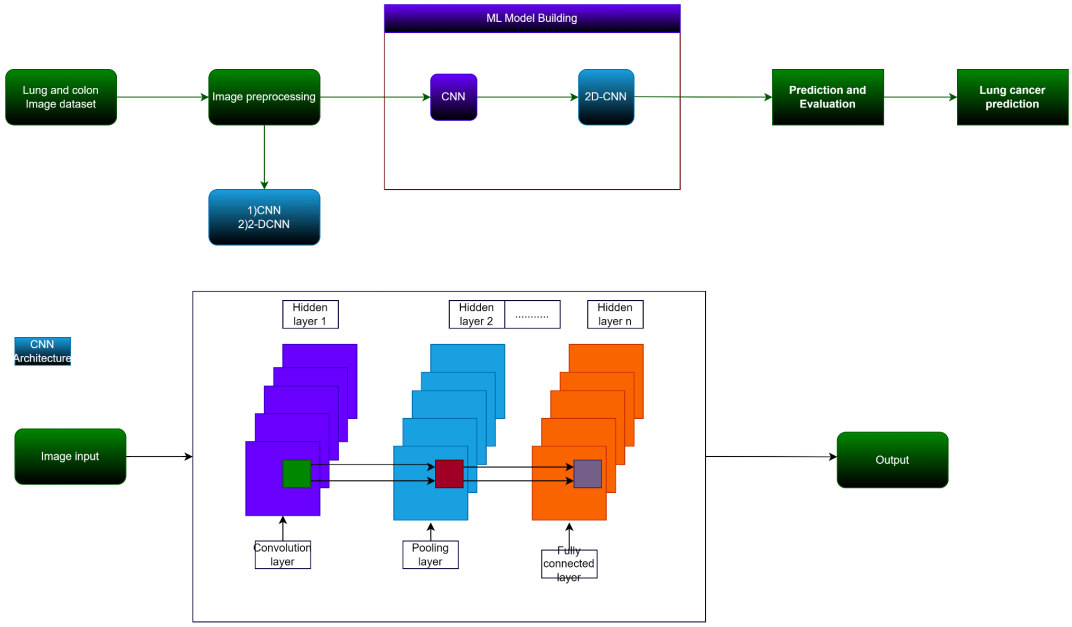

(3)


(4)


(5)


(6)

3.2.2 Architecture



(7)

3.2.3 Machine Learning Algorithm used

We have used both DeepLearning and Machine Learning algorithms such as Convolutional Neural Networks(CNN), 2D-Convolutional Neural Networks.

Convolution Neural Networks(CNN):

CNN is used to image analysis tasks. They will learn from raw pixel data and they will capture complex spatial patterns in medical imaging.

2D-CNN:-

It is a set of medical images(Histopathological images) which represents different patients. It will handle the characteristics of medical images like varying resolutions, complex anatomical structures, it will involve multiple convolution layers, pooling layers and fully connected layers.

3.2.4 Model Selection and Evaluation

Model Selection:

Convolutional Neural Network(CNN) and 2D-CNN is mainly used for medical images due to its automatically learn hierarchical features from image data. It have pre-trained CNN architectures like ResNet which have been trained on large-scale image dataset like ImageNet. This CNN

and 2D-CNN experimenting with different network architectures which includes convolution, filter, pooling and fully connected layers.Then in 2D-CNN architecture we need to perform hyperparameter tuning to optimize its performance.

Model Evaluation:

Firstly we need to split the medical image dataset into training, validation and testing sets. The sets is used to train, tune hyperparameters and it is used to evaluate the final performance. Then, choose a appropriate performance metrics for evaluating the model performance. It includes accuracy, precision, recall, F1-score. If the dataset is limited then use techniques like K-fold cross validation for accessing the model's generation performance. To optimize its performance tune the hyperparameters of the model.

Hyperparameters includes learning rate, batch size.Then we must validate the trained model to assess its generation ability in real-world situations.

Chapter 4
Result and Discussion

We have used histopatical images from Lung and Colon dataset and we have used Deep Learning algorithms. We have tested the code and get accuracy of 93% and the total losses are 21% and still we are trying include the Machine Learning algorithms and few more presided Deep Learning algorithm techniques later on our project and the better accuracy while comparing to this algorithm. We have used CNN and 2D-CNN for training the process and splited the dataset into training and testing after that we have divided into cancerous and non cancerous images. The exiting system has the accuracy of 96% which we have taken same dataset and executed in the workspace of kaggle notebook.

Loss Calculation:

$$Train\ Loss = Loss\ Function(y_{true},\ y_{pred})$$

$$Validation\ Loss = Loss\ Function(y_{true},\ y_{pred})$$

$$Test\ Loss = Loss\ Function(y_{true}, y_{pred})$$

Accuracy Calculation:

$$Training\ Accuracy = \frac{Total\ Training\ Samples}{Correctly\ Predicted\ Samples} \times 100\,\%$$

$$Validation\ Accuracy = \frac{Total\ Validating\ Samples}{Correctly\ Predicted\ Samples} \times 100\,\%$$

$$Testing\ Accuracy = \frac{Total\ Testing\ Samples}{Correctly\ Predicted\ Samples} \times 100\,\%$$

Chapter 5
Conclusion and Future Scope

Future Scope:-

Exploring the advance image processing and also analyze the techniques such as deep learning to improve the detection of accuracy and efficiency.By fusion of multiple imaging modalities such as MRI and clinical data such as patient history to develop diagnostic models.Focus on enhancing model interpretability and explainability to facilitate adoption by healthcare practitioners and build trust. Collab with healthcare institutions and research center to conduct model validation for large scale studies.

Conclusion:

The applications of medical images it witnessed many advancements and holds immense potential for healthcare .In future the research effort should be on the improving the existing model to face the key challenges. Effects towards ensuring model interpretability and scalability will be essential to trust and accept within the medical community.

Chapter 6
References

1. Saeedi, S., Rezayi, S., Keshavarz, H., & R. Niakan Kalhori, S. (2023). MRI-based brain tumor detection using convolutional deep learning methods and chosen machine learning techniques. BMC Medical Informatics and Decision Making, 23(1), 16.
2. Rana, M., & Bhushan, M. (2023). Machine learning and deep learning approach for medical image analysis: diagnosis to detection. Multimedia Tools and Applications, 82(17), 26731-26769.
3. Safdar S, Rizwan M, Gadekallu TR, Javed AR, Rahmani MKI, Jawad K, Bhatia S. Bio-Imaging-Based Machine Learning Algorithm for Breast Cancer Detection. *Diagnostics*. 2022; 12(5):1134.
4. Sharma, A., Yadav, D. P., Garg, H., Kumar, M., Sharma, B., & Koundal, D. (2021). Bone cancer detection using feature extraction based machine learning model. Computational and Mathematical Methods in Medicine, 2021.
5. Hirra, I., Ahmad, M., Hussain, A., Ashraf, M. U., Saeed, I. A., Qadri, S. F., ... & Alfakeeh, A. S. (2021). Breast cancer classification from histopathological images using patch-based deep learning modeling. IEEE Access, 9, 24273-24287.

Appendix-I

```
import os
import time
import shutil
import pathlib
import itertools
from sklearn.model_selection import train_test_split
from PIL import Image
# import data handling tools
import cv2
import numpy as np
import pandas as pd
import seaborn as sns
sns.set_style('darkgrid')
import matplotlib.pyplot as plt
from sklearn.model_selection import train_test_split
from sklearn.metrics import confusion_matrix, classification_report
# import Deep learning Libraries
import tensorflow as tf
from tensorflow import keras
from tensorflow.keras.models import Sequential
from tensorflow.keras.optimizers import Adam, Adamax
from tensorflow.keras.preprocessing.image import ImageDataGenerator
```

```
from tensorflow.keras.layers import Conv2D, MaxPooling2D, Flatten, Dense,
Activation, Dropout, BatchNormalization

from tensorflow.keras import regularizers

# Ignore Warnings

import warnings

warnings.filterwarnings("ignore")

print ('modules loaded')

"""from os.path import exists

def is_file_imported_from_drive(file_name):

  Checks if a file is imported from Google Drive or not.

  Args:

   file_name: The name of the file to check.

  Returns:

    True if the file is imported from Google Drive, False otherwise.

  drive_mount_path = "/content/drive/My Drive"

  file_path = os.path.join(drive_mount_path, file_name)

  return exists(file_path)"""

import os

file_name =
'/kaggle/input/lung-and-colon-cancer-histopathological-images/lung_colon_image_set'

file_paths = []

labels = []

# Traverse through directory structure to collect file paths and labels

folds = os.listdir(file_name)

for fold in folds:

    sub_fold_path = os.path.join(file_name, fold)

    sub2folds = os.listdir(sub_fold_path)
```

```
    for c_fold in sub2folds:
        c_sub_path = os.path.join(sub_fold_path, c_fold)
        subcfolds = os.listdir(c_sub_path)
        for filepath in subcfolds:
            file_path = os.path.join(c_sub_path, filepath)
            file_paths.append(file_path)
            labels.append(c_fold)
# Create dataframe from file paths and labels
fseries = pd.Series(file_paths, name='filepath')
lseries = pd.Series(labels, name='labels')
df = pd.concat([fseries, lseries], axis=1)
set(labels)
"file_paths"
 'df'
train_df, ts_df = train_test_split(df, test_size = 0.2 , random_state = 50, stratify = df['labels'])
valid_df, test_df = train_test_split(ts_df, test_size = 0.5 , random_state = 40, stratify  =
ts_df['labels'])
'train_df'
'valid_df'
'test_df'
## import matplotlib.pyplot as plt
# Function to plot label distribution as line plot
def plot_label_distribution_line(dataframe, title):
    plt.figure(figsize=(10, 6))
    label_counts = dataframe['labels'].value_counts().sort_index()
    plt.plot(label_counts.index, label_counts.values, marker='o', linestyle='-', color='b')
```

```
    plt.title(title)
    plt.xlabel('Labels')
    plt.ylabel('Count')
    plt.xticks(rotation=45)
    plt.grid(True)
    plt.show()
# Plot label distribution for Train DataFrame
plot_label_distribution_line(train_df, 'Train Data Label Distribution')
# Plot label distribution for Test DataFrame
plot_label_distribution_line(test_df, 'Test Data Label Distribution')
# Plot label distribution for Validation DataFrame
plot_label_distribution_line(valid_df, 'Validation Data Label Distribution')
batch_size = 64
img_size =  (224,224)
geny = ImageDataGenerator()
train_geny = geny.flow_from_dataframe(train_df, x_col = 'filepath', y_col = 'labels', target_size=
img_size , batch_size = batch_size, shuffle = True , class_mode = 'categorical', color_mode =
'rgb' )
valid_geny = geny.flow_from_dataframe(valid_df, x_col = 'filepath', y_col = 'labels', target_size=
img_size , batch_size = batch_size, shuffle = True , class_mode = 'categorical', color_mode =
'rgb' )
test_geny = geny.flow_from_dataframe(test_df, x_col = 'filepath', y_col = 'labels', target_size=
img_size , batch_size = batch_size, class_mode = 'categorical', color_mode = 'rgb' )
from keras.models import Sequential
from keras.layers import Conv2D, MaxPooling2D, Flatten, Dense
from keras.optimizers import Adamax
model = Sequential([
    Conv2D(filters=64, kernel_size=(3, 3), activation="relu", padding="same", input_shape=(224,
224, 3)),
```

```
    Conv2D(filters=64, kernel_size=(3, 3), padding="same", activation="relu"),
    MaxPooling2D((2, 2)),
    Conv2D(filters=128, kernel_size=(3, 3), padding="same", activation="relu"),
    Conv2D(filters=128, kernel_size=(3, 3), padding="same", activation="relu"),
    MaxPooling2D((2, 2)),
    Conv2D(filters=256, kernel_size=(3, 3), padding="same", activation="relu"),
   Conv2D(filters=256, kernel_size=(3, 3), padding="same", activation="relu"),
    MaxPooling2D((2, 2)),
    Conv2D(filters=512, kernel_size=(3, 3), padding="same", activation="relu"),
    Conv2D(filters=512, kernel_size=(3, 3), padding="same", activation="relu"),
    Conv2D(filters=512, kernel_size=(3, 3), padding="same", activation="relu"),
    MaxPooling2D((2, 2)),
    Conv2D(filters=512, kernel_size=(3, 3), padding="same", activation="relu"),
    Conv2D(filters=512, kernel_size=(3, 3), padding="same", activation="relu"),
    Conv2D(filters=512, kernel_size=(3, 3), padding="same", activation="relu"),
    Flatten(),
    Dense(256, activation='relu'),
    Dense(128, activation='relu'),
    Dense(5, activation='softmax') # Adjusted to 5 output classes to match your target labels
])
# Compile the model
model.compile(optimizer=Adamax(learning_rate=0.001), loss='categorical_crossentropy',
metrics=['accuracy'])
model.summary()
history=  model.fit(train_geny, epochs= 5,verbose= 1 ,validation_data = valid_geny,
validation_steps= None, shuffle= False  )
# Assuming `model` is your trained model
```

```
y_pred = model.predict(test_geny)

y_pred_class = np.argmax(y_pred, axis=1)

# Calculate the number of steps for evaluation

ts_length = len(test_df)

test_batch_size = max(sorted([ts_length // n for n in range(1, ts_length + 1) if ts_length % n == 0
and ts_length / n <= 80]))

test_steps = ts_length // test_batch_size

# Evaluate the model on the training, validation, and test datasets

train_score = model.evaluate(train_geny, steps=test_steps, verbose=1)

valid_score = model.evaluate(valid_geny, steps=test_steps, verbose=1)

test_score = model.evaluate(test_geny, steps=test_steps, verbose=1)

# Print the evaluation results

print("Train Loss: ", train_score[0] * 100)

print("Train Accuracy: ", train_score[1] * 100)

print('-' * 20)

print("Validation Loss: ", valid_score[0] * 100)

print("Validation Accuracy: ", valid_score[1] * 100)

print('-' * 20)

print("Test Loss: ", test_score[0] * 100)

print("Test Accuracy: ", test_score[1] * 100)
```

Appendix-II