**Procedure to run the application**

1. Load the data (LC25000), please follow the same directory structure as shared.
2. Performed data augmentation and color transformation with colortransformation.ipynb file, followed by post processing.
3. After normalization process, classification framework is established using FastAI-2, and optimal hyperparameters are stored.
4. Modified transfer learning module (ResNet34, VGG11) are applied to access the outcome of proposed system.
5. Every file contains performance evaluation matrix within these are summarized later on.
6. All the colab files are shared.

**Resnet34\_lung\_cancer**

import torch

print(torch.\_\_version\_\_)

# Commented out IPython magic to ensure Python compatibility.

# %matplotlib inline

import os

import numpy as np

import pandas as pd

import matplotlib.pyplot as plt

import matplotlib.patches as patches

from sklearn.metrics import roc\_curve, auc

!pip install fastai

from fastai.vision.all import cnn\_learner, RandTransform, ImageDataLoaders, ClassificationInterpretation, models

from fastai.metrics import error\_rate, accuracy

image\_directory = '/content/drive/MyDrive/LC25000/'

data = ImageDataLoaders.from\_folder(image\_directory, train='train', valid='val', size=224, bs=32, num\_workers=8)

data.show\_batch()

print(len(data.train\_ds), len(data.valid\_ds))

dir(models)

learn = cnn\_learner(data, models.resnet34, metrics = [accuracy])

learn.summary()

learn.unfreeze()

learn.lr\_find()

learn.fit\_one\_cycle(5,lr\_max= slice(1.4e-4,1.25e-3))

learn.recorder.plot\_loss(skip\_start=0, with\_valid=True)

learn.save('/content/drive/MyDrive/LC25000/models/lung\_cancer\_classifier')

interp = ClassificationInterpretation.from\_learner(learn)

interp.plot\_top\_losses(9, figsize=(12,12))

interp.plot\_confusion\_matrix(figsize=(6,6), dpi=300)

interp.print\_classification\_report()

preds,y, loss = learn.get\_preds(with\_loss=True)

# get accuracy

acc = accuracy(preds, y)

print('The accuracy is {0} %.'.format(acc))

# probs from preds

probs = np.exp(preds[:,1])

# Compute ROC curve

fpr, tpr, thresholds = roc\_curve(y, probs, pos\_label=1)

# Compute ROC area

roc\_auc = auc(fpr, tpr)

print('ROC area is {0}'.format(roc\_auc))

plt.figure()

plt.plot(fpr, tpr, color='darkorange', label='ROC curve (area = %0.2f)' % roc\_auc)

plt.plot([0, 1], [0, 1], color='navy', linestyle='--')

plt.xlim([-0.01, 1.0])

plt.ylim([0.0, 1.01])

plt.xlabel('False Positive Rate')

plt.ylabel('True Positive Rate')

plt.title('Receiver operating characteristic')

plt.legend(loc="lower right")

**Vgg11\_lung\_cancer**

# Commented out IPython magic to ensure Python compatibility.

# %matplotlib inline

import os

import numpy as np

import pandas as pd

import matplotlib.pyplot as plt

import matplotlib.patches as patches

from sklearn.metrics import roc\_curve, auc

import torch

print(torch.\_\_version\_\_)

from google.colab import drive

drive.mount('/content/drive')

from fastai.vision.all import cnn\_learner, RandTransform, ImageDataLoaders, ClassificationInterpretation, models

from fastai.metrics import error\_rate, accuracy

image\_directory = '/content/drive/MyDrive/LC25000/'

data = ImageDataLoaders.from\_folder(image\_directory, train='train', valid='val', size=224, bs=16, num\_workers=8)

print(len(data.train\_ds), len(data.valid\_ds))

dir(models)

learn = cnn\_learner(data, models.vgg11, metrics = [accuracy])

learn.summary()

learn.unfreeze()

learn.lr\_find()

learn.fit\_one\_cycle(5,lr\_max=0.00083)

learn.recorder.plot\_loss(skip\_start=0, with\_valid=True)

learn.save('/content/drive/MyDrive/LC25000/models/lung\_cancer\_classifier')

interp = ClassificationInterpretation.from\_learner(learn)

interp.plot\_top\_losses(9, figsize=(12,12))

interp.plot\_confusion\_matrix(figsize=(6,6), dpi=60)

interp.print\_classification\_report()

preds,y, loss = learn.get\_preds(with\_loss=True)

# get accuracy

acc = accuracy(preds, y)

print('The accuracy is {0} %.'.format(acc))

# probs from preds

probs = np.exp(preds[:,1])

# Compute ROC curve

fpr, tpr, thresholds = roc\_curve(y, probs, pos\_label=1)

# Compute ROC area

roc\_auc = auc(fpr, tpr)

print('ROC area is {0}'.format(roc\_auc))

plt.figure()

plt.plot(fpr, tpr, color='darkorange', label='ROC curve (area = %0.2f)' % roc\_auc)

plt.plot([0, 1], [0, 1], color='navy', linestyle='--')

plt.xlim([-0.01, 1.0])

plt.ylim([0.0, 1.01])

plt.xlabel('False Positive Rate')

plt.ylabel('True Positive Rate')

plt.title('Receiver operating characteristic')

plt.legend(loc="lower right")