```
close all


%% Data Prep
% Training set
folderPath = 'train';
imageFiles = dir(fullfile(folderPath, '*.jpg'));
numImages = length(imageFiles);

for i = 1:numImages
   imageFilename = fullfile(folderPath, imageFiles(i).name);
   image = imread(imageFilename);
   trainData(i).image = image;
end
disp(trainData)
% Specify the path to the folder containing your XML files
xmlFilesFolder = 'train\';
% List all XML files in the folder
xmlFiles = dir(fullfile(xmlFilesFolder, '*.xml'));
% Initialize arrays to store bounding box information
allBboxes = [];
allLabels = {};
% Loop through each XML file
for i = 1:numel(xmlFiles)
   % Read the XML file
   xmlFile = fullfile(xmlFilesFolder, xmlFiles(i).name);
   xmlData = xml2struct(xmlFile);
   % Extract bounding box information
   annotations = xmlData.annotation;

   % Check if there is more than one object annotation
   if iscell(annotations.object)
       for j = 1:numel(annotations.object)
           % Extract label
           label = annotations.object{j}.name.Text;
           % Extract bounding box coordinates
           xmin = str2double(annotations.object{j}.bndbox.xmin.Text);
           ymin = str2double(annotations.object{j}.bndbox.ymin.Text);
           xmax = str2double(annotations.object{j}.bndbox.xmax.Text);
           ymax = str2double(annotations.object{j}.bndbox.ymax.Text);
           % Store bounding box information
           allBboxes = [allBboxes; xmin, ymin, xmax - xmin, ymax - ymin];  %
[x, y, width, height]
           allLabels = [allLabels; label];
       end
   else
       % Extract label
       label = annotations.object.name.Text;
       % Extract bounding box coordinates
       xmin = str2double(annotations.object.bndbox.xmin.Text);
       ymin = str2double(annotations.object.bndbox.ymin.Text);
       xmax = str2double(annotations.object.bndbox.xmax.Text);
       ymax = str2double(annotations.object.bndbox.ymax.Text);
       % Store bounding box information
       allBboxes = [allBboxes; xmin, ymin, xmax - xmin, ymax - ymin]; % [x, y,
width, height]
       allLabels = [allLabels; label];
   end

end
% Get a list of all files in the folder
files = dir(fullfile(folderPath, '*.xml'));  % Modify the extension based on
your image format
imageFilesNames = {files.name};
uniqueImages = unique(imageFilesNames); % Assuming imageFileNames is a cell
array of image file names
numImages = numel(uniqueImages);
boxCellArray = cell(numImages, 1);
labelCellArray = cell(numImages, 1);
for i = 1:numImages
   % Find indices corresponding to the current image
   indices = strcmp(imageFilesNames, uniqueImages{i});

   % Extract bounding box data and labels for the current image
   currentBboxes = allBboxes(indices, :);
   currentLabels = allLabels(indices);

   % Store data in cell arrays
   boxCellArray{i} = currentBboxes;
   labelCellArray{i} = currentLabels;
end
% Create a table with the required columns
boxLabelsTable = table(boxCellArray, labelCellArray, 'VariableNames',
{'BoundingBox', 'Label'});
% Create a boxLabelDatastore
boxLabelDS = boxLabelDatastore(boxLabelsTable);
% Initialize imageDatastore
imds = imageDatastore(fullfile(folderPath, {imageFiles.name}));
% Create a boxLabelDatastore
boxLabelDS = boxLabelDatastore(boxLabelsTable);
trainingData = (imds,boxLabelDS);
disp(trainingData)
%% Model Arcitecture
net = resnet50;
imageSize = [416 416 3];
numClasses = 3;
anchorBoxes = [16, 16; 24, 24; 32, 32];
featureLayer = 'activation_49_relu';
```

```matlab
lgraph = fasterRCNNLayers(imageSize,numClasses,anchorBoxes,net,featureLayer);
%% Training
% Step 7: Train the Faster R-CNN model
checkpointDir = 'C:\Users\mpetr\Downloads\BCC.v4-416x416_aug.voc\CheckPoint\';
% Create the checkpoint directory if it doesn't exist
if ~exist(checkpointDir, 'dir')
   mkdir(checkpointDir);
end
options = trainingOptions('sgdm', ...
     'MiniBatchSize', 1,...
     'MaxEpochs', 1, ...
     'InitialLearnRate', 1e-4, ...
     'LearnRateSchedule', 'piecewise', ...
     'LearnRateDropFactor', 0.1, ...
     'LearnRateDropPeriod', 3,...
     'CheckpointPath', 'CheckPoint','Verbose',true);
detector = trainFasterRCNNObjectDetector(trainingData, lgraph, options);
checkpointFile = 'detector_checkpoint.mat';
save(fullfile(checkpointDir, checkpointFile), 'detector');
trainedDetector = load(fullfile(fullfile(checkpointDir, checkpointFile)));
detectors = trainedDetector.detector;
% Step 8: Test the model on new images
% Use the trained model to perform object detection on new images
testImage =
imread('test\BloodImage_00359_jpg.rf.03331bc3903822adbe982fe9e7cd061b.jpg');
resizedTestImage = imresize(testImage, imageSize(1:2));
[bbox, scores, labels] = detect(detectors, resizedTestImage);

disp(imageSize)
% Step 9: Visualize the results
% Display the original image with bounding boxes around detected objects
detectedImg = insertObjectAnnotation(testImage, 'Rectangle', bbox,
LineWidth=5);
disp(bbox)
disp(labels)
imshow(detectedImg)
%% Step 10
% Model Evaluation
groundTruthData = [];
for i = 1:length(boxLabelDS)
   groundTruthData{i} = boxLabelDS.read(i);
end
truePositives = [];
falsePositives = [];
falseNegatives = [];
ious = [];
for i = 1:length(testImages)
   % Load test image
   testImage = imread(testImages{i});
   % Apply Faster R-CNN model to detect objects
   [bboxes, scores, labels] = detect(fasterRCNNModel, testImage);
   % Match predicted bounding boxes with ground truth bounding boxes
   groundTruthBoxes = groundTruthData{i}.boundingBoxes;
   matchedBoxes = matchBoundingBoxes(bboxes, groundTruthBoxes);
   % Calculate TP, FP, FN, and IoU
   tp = sum(matchedBoxes(:, 3));
   fp = sum(matchedBoxes(:, 1)) - tp;
   fn = sum(groundTruthBoxes(:, 2)) - tp;
   matchedIous = matchedBoxes(:, 4);
   truePositives = [truePositives; tp];
   falsePositives = [falsePositives; fp];
   falseNegatives = [falseNegatives; fn];
   ious = [ious; matchedIous];
end
% Calculate precision, recall, F1 score, and mean IoU
precision = truePositives / (truePositives + falsePositives);
recall = truePositives / (truePositives + falseNegatives);
f1Score = 2 * precision * recall / (precision + recall);
meanIoU = mean(ious);
% Calculate mean average precision (mAP)
map = calculateMAP(groundTruthData, testImages, bboxes, labels);
meanAP = mean(map);
% Display evaluation results
disp(['Precision: ', num2str(precision)]);
disp(['Recall: ', num2str(recall)]);
disp(['F1 Score: ', num2str(f1Score)]);
disp(['Mean IoU: ', num2str(meanIoU)]);
disp(['Mean AP: ', num2str(meanAP)]);
%% Step 11: Results
% Get Ratio of WBC and RBC
% Extract the first two labels
firstTwoLabels = labels(2:3);
% Count the number of times each label appears
uniqueLabels = unique(firstTwoLabels);
counts = zeros(length(uniqueLabels), 1);
for i = 1:length(firstTwoLabels)
   for j = 1:length(uniqueLabels)
       if strcmp(firstTwoLabels(i), uniqueLabels(j))
           counts(j) = counts(j) + 1;
       end
   end
end
% Calculate the ratio between the first two labels
ratio = counts(1) / counts(3);
% Print the results
if ratio < 6 %Values minimized for available data
   disp("Chance of AML: LOW");
elseif ratio < 10
```

```
   disp("Chance of AML: MEDIUM");
else
   disp("Chance of AML: HIGH");
end
```