COVID-19 Detection Usign X-Rays

by Joshua Rothe

# Image pre-processing, K-Means

K = 10; % for K-means

% Class - normal images

num\_images = 2000; % Number of images in train dataset is 1-8000

normal\_img = zeros((299\*299),num\_images); % Initialize empty array

for i = 1:num\_images

filename = sprintf('./data/normal/Normal-%i.png', i);

img = (imread(filename));

% Perform K-means for the image (comment out below if not using)

% img = imsegkmeans(img,K);

normal\_img(:,i) = double(reshape(img,[],1)); % reshapes image as column vector

end % and adds it to normal\_img

labels(1:num\_images,1) = "normal";

imshow(img);

% Class - COVID

num\_images2 = 1000; % Number of images in training dataset is 1-3000

cov\_img = zeros((299\*299),num\_images2); % Initialize empty array

for i = 1:num\_images2

filename = sprintf('./data/COVID/COVID-%i.png', i);

img = (imread(filename));

% Perform K-means for the image (comment out below if not using)

% img = imsegkmeans(img,K);

cov\_img(:,i) = double(reshape(img,[],1)); % reshapes image as column vector

end % and adds it to cov\_img

labels(num\_images+1:num\_images2+num\_images,1) = "COVID";

imshow(img);

% Combine images and labels

train\_images = [normal\_img, cov\_img];

train\_labels = labels;

# Apply PCA

%% Train and find the PCA dimensions of the training data

% First reduce the training data to a computationally manageable size

% (note: in this instance, size is unchanged, but may adjust as needed)

train\_size = num\_images + num\_images2;

base = 50; % Base size, given

% Reduced variables are for if we reduce the number if pics taken from

% dataset.

train\_images\_reduced = train\_images(:,[1:train\_size]);

train\_labels\_reduced = train\_labels([1:train\_size],:);

% Normalize the data, will be calling it X for simplicity

X = train\_images\_reduced;

Xmean = mean(X); % mean of X

Xnorm = X - Xmean; % normalized

% Calculate covariance matrix

covar\_matrix = (Xnorm' \* Xnorm) / (train\_size - 1);

% Calculate Eigenvectors and Eigenvalues of Covariance Matrix

[Xeivec, Xeival] = eig(covar\_matrix);

% Now project the dataset

proj\_train = Xnorm \* Xeivec;

% Back into larger data format

Xnormk\_train = proj\_train \* Xeivec';

# Create Models

% Create a model with standard params

Mdl = fitcecoc(Xnormk\_train',train\_labels\_reduced');

% Note: This model performs poorly. More than 50% error when

% tested using predict().

% error = resubLoss(Mdl) % resubloss error

% Obtain better model performance with parameters

t = templateSVM('Standardize',true);

Mdl2 = fitcecoc(Xnormk\_train',train\_labels\_reduced',...

'Learners',t,'ClassNames',{'normal','COVID'});

% options = statset('UseParallel',true);

% CVMdl2 = crossval(Mdl2,'Options',options);

% kFoldError2 = kfoldLoss(CVMdl2) % kfoldloss error from cross validated model

% Create model with optimized parameters

t = templateSVM('Standardize',true,'KernelScale', 107.7, ...

'BoxConstraint', .0010908);

Mdl3 = fitcecoc(Xnormk\_train',train\_labels\_reduced',...

'Learners',t,'Coding',"onevsall",'ClassNames',{'normal','COVID'});

% options = statset('UseParallel',true);

% CVMdl3 = crossval(Mdl3,'Options',options);

% kFoldError3 = kfoldLoss(CVMdl3) % kfoldloss error from cross validated model

# Test Models

% Return a vector of predicted values, should all be non-COVID.

% test data is 8001 - 8100

clear test\_label\_healthy2; clear test\_data\_healthy;

clear test\_label\_healthy3; k = 1;

%test\_data\_healthy = zeros((299\*299),(8100-8001+1)); % Initialize empty array

for i = 7001:8000

filename = sprintf('./data/normal/Normal-%i.png', i);

img = (imread(filename));

test\_data\_healthy(:,k) = double(reshape(img,[],1)); % reshapes image as column vector

k = k + 1; % increment k for matrix pos

end

[test\_label\_healthy2] = predict(Mdl2,test\_data\_healthy');

[test\_label\_healthy3] = predict(Mdl3,test\_data\_healthy');

% Return a vector of predicted values, should all be all COVID infections.

% test data is 2001 - 3000

clear test\_label\_covid2; clear test\_data\_covid;

clear test\_label\_covid3; k = 1;

%test\_data\_covid = zeros((299\*299),(3101-3001+1)); % Initialize empty array

for i = 2001:3000

filename = sprintf('./data/COVID/COVID-%i.png', i);

img = (imread(filename));

test\_data\_covid(:,k) = double(reshape(img,[],1)); % reshapes image as column vector

k = k + 1; % increment k for matrix pos

end

[test\_label\_covid2] = predict(Mdl2,test\_data\_covid');

[test\_label\_covid3] = predict(Mdl3,test\_data\_covid');

## Mdl2 - Default Paramaters

Accuracy with both the Healthy and COVID datasets was observed to try and see if the model skewed towards false positives or false negatives.

### Healthy Test

% Check performance by counting number of times correct value was reached

perf\_healthy\_temp2a = categorical(test\_label\_healthy2);

labels\_perf2a = categories(perf\_healthy\_temp2a) % displays labels

perf\_healthy2a = countcats(perf\_healthy\_temp2a) % displays occurance per label

Mdl2 with no K-Means is 93.0% accurate at detecting healthy images.

With K = 3 it is 72.8% accurate.

With K = 5 it is 69.8% accurate.

With K = 10 it is 92.4% accurate.

### COVID Test

% Check performance by counting number of times correct value was reached

perf\_covid\_temp2b = categorical(test\_label\_covid2);

labels\_perf2b = categories(perf\_covid\_temp2b) % displays labels

perf\_covid2b = countcats(perf\_covid\_temp2b) % displays occurance per label

Mdl2 with no K-Means is 70.9% accurate at detecting COVID infection.

With K = 3 it is 89.2% accurate.

With K = 5 it is 70.4% accurate.

With K = 10 it is 89.7% accurate.

## Mdl3 - Optimized Paramaters

Same as Mdl2 except with the optimized model. Results are greatly improved.

### Healthy Test

% Check performance by counting number of times correct value was reached

perf\_healthy\_temp3a = categorical(test\_label\_healthy3);

labels\_perf3a = categories(perf\_healthy\_temp3a) % displays labels

perf\_healthy3a = countcats(perf\_healthy\_temp3a) % displays occurance per label

Mdl2 with no K-Means is 99.8% accurate at detecting healthy images.

With K = 3 it is 88.4% accurate.

With K = 5 it is 100.0% accurate.

With K = 10 it is 99.7% accurate.

### COVID Test

% Check performance by counting number of times correct value was reached

perf\_covid\_temp3b = categorical(test\_label\_covid3);

labels\_perf3b = categories(perf\_covid\_temp3b) % displays labels

perf\_covid3b = countcats(perf\_covid\_temp3b) % displays occurance per label

Mdl3 with no K-Means is 100.0% accurate at detecting COVID infection.

With K = 3 it is 93.0% accurate.

With K = 5 it is 99.7% accurate.

With K = 10 it is 99.5% accurate.