from keras.models import load\_model

#preds = [model.predict(X\_test) for model in models]

pred\_threshold = 0.5

pred1 = model1.predict(train\_images)

pred2 = model2.predict(train\_images)

pred4 = model4.predict(train\_images)

pred1\_b = np.where(pred1[:,:,:,:] > pred\_threshold, 1, 0)

pred2\_b = np.where(pred2[:,:,:,:] > pred\_threshold, 1, 0)

pred4\_b = np.where(pred4[:,:,:,:] > pred\_threshold, 1, 0)

#Grid search for the best combination of w1, w2, w3 that gives maximum acuracy

import pandas as pd

df = pd.DataFrame([])

models = [ model1, model2, model4]

preds=np.array([ pred1, pred2, pred4])

union\_threshold = 0.99

overlap\_threshold = 0.5

for w1 in range(0, 10):

for w2 in range(0, 10):

for w3 in range(0, 10):

wts = [w1/10.,w2/10.,w3/10.]

wted\_preds = np.tensordot(preds, wts, axes=((0),(0)))

#wted\_ensemble\_pred = np.argmax(wted\_preds, axis=3)

if(w1+w2+w3 > 9):

overlap = train\_masks[:,:,:,:] \* wted\_preds[:,:,:,:] # Logical AND

overlap = np.where(overlap[:,:,:,:] > overlap\_threshold, 1, 0)

union = train\_masks[:,:,:,:] + wted\_preds[:,:,:,:] # Logical OR

union = np.where(union[:,:,:,:] > union\_threshold, 1, 0)

IOU = overlap.sum()/float(union.sum())

print("Now predciting for weights :", w1/10., w2/10., w3/10., " : IOU = ", IOU)

df = df.append(pd.DataFrame({'wt1':wts[0],'wt2':wts[1],

'wt3':wts[2], 'IOU': IOU}, index=[0]), ignore\_index=True)

max\_iou\_row = df.iloc[df['IOU'].idxmax()]

print("Max IOU of ", max\_iou\_row[3], " obained with w1=", max\_iou\_row[0],

" w2=", max\_iou\_row[1], " and w3=", max\_iou\_row[2])

#############################################################

Now predciting for weights : 0.5 0.0 0.7 : IOU = 0.6721078398497753

Now predciting for weights : 0.5 0.0 0.8 : IOU = 0.6811701575315056

..

..

..

Now predciting for weights : 0.9 0.9 0.8 : IOU = 0.7015908095823864

Now predciting for weights : 0.9 0.9 0.9 : IOU = 0.6997320371408986

Max IOU of 0.7288750860641929 obained with w1= 0.6 w2= 0.9 and w3= 0.4

wts = [0.6,0.9,0.4]

preds=np.array([ pred1, pred2, pred4])

wted\_preds = np.tensordot(preds, wts, axes=((0),(0)))

for i in range(len(wted\_preds)):

weighted\_preds = np.where(wted\_preds[:,:,:,:] > 0.5, 1, 0)

n = 1

plt.subplot(231)

prediction = weighted\_preds[n]

prediction = prediction[:,:,0]

plt.title('Testing Image')

plt.imshow(train\_images[n])

plt.imshow(prediction, cmap='jet', alpha=0.5)

#plt.text(10, 10, 'Cancer', bbox=dict(fill=False, edgecolor='red', linewidth=2))

#plt.savefig("Testing image.png")

#im.save(d+f"{i}.jpeg")

plt.subplot(232)

plt.title('Testing Label')

plt.legend(['Yellow = Legion', 'Purple = Non-Legion'])

ground\_truth=train\_masks[n]

gt=ground\_truth[:,:,0]

plt.imshow(gt)

#plt.savefig("Testing labe.png")

plt.subplot(233)

plt.title('Prediction')

prediction = weighted\_preds[n]

prediction = prediction[:,:,0]

plt.imshow(prediction)

n = n+1