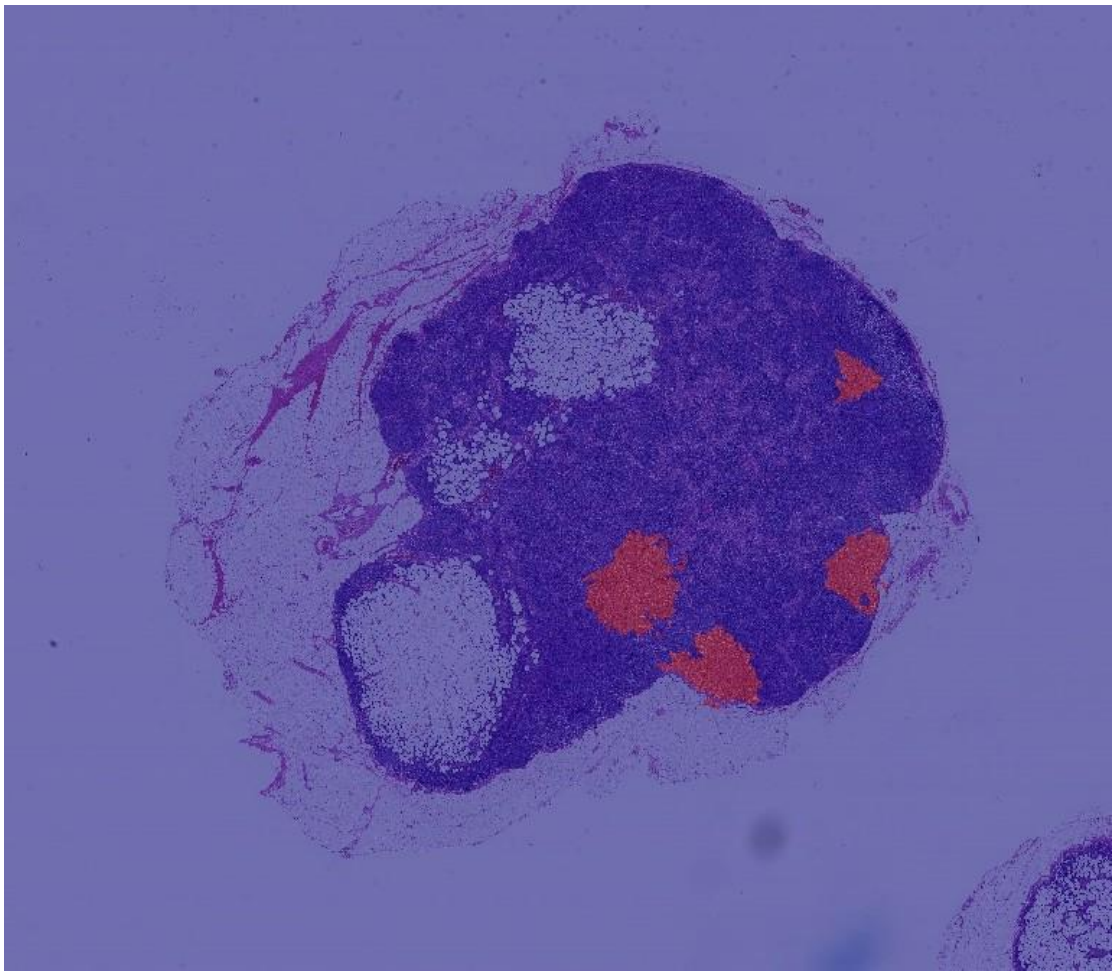
Tumor Detection on Gigapixel Pathology Images

W4995 Course Project

Supplementary Report



Prepared by Hongyu Li (hl3099) and Fan Yang (fy2232)

Contents

1 Workflow

In our project, we used *tumor_091.tif* as our training data. This slide has 4 main tumor areas which is shown as red regions in Fig 1-1.

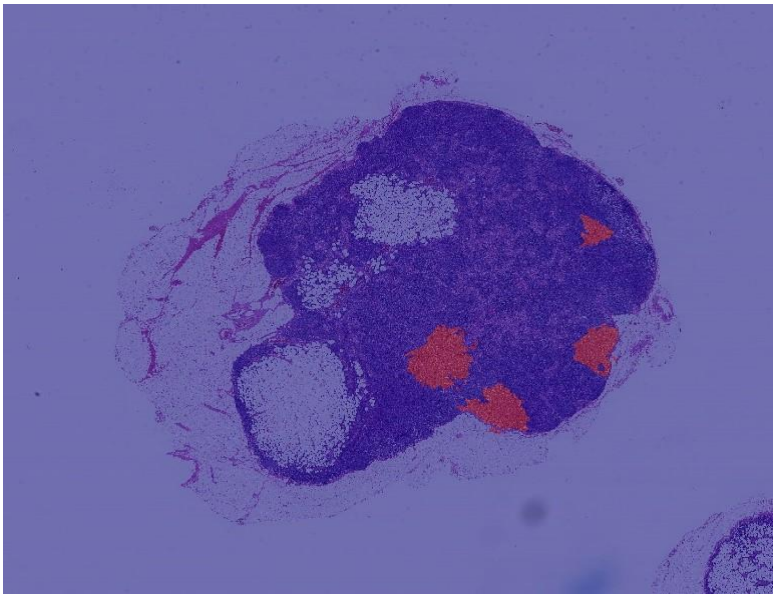


Fig 1-1: Tumor 091 with Cancer Mask

And our pipeline is shown as follows:

1. **Step 0: Download Slide and Mask Image**. Here, we downloaded *tumor_091.tif* and *tumor_mask_091.tif*. This slide has eight levels and the dimensions for each level are shown in Table 1-1.

Table 1-1: Details of Tumor 091

Level	Dimensions	Down-sample Factor
0	(61440, 53760)	1
1	(30720, 26880)	2
2	(15360, 13440)	4
3	(7680, 6720)	8
4	(3840, 3360)	16
5	(1920, 1680)	32
6	(960, 840)	64
7	(480, 420)	128

2. **Step 1: Read Slide and Preprocess Data**. In this step, we firstly extract patches and labels from the slide at **level 3**. And since our data are extremely imbalanced (tissue regions are greatly larger than tumor regions), we processed rescaling,

resampling and augmentations for our dataset. There are some parameters needed to be tuned in this step. The details about tuning is shown in Sec 2.

3. **Step 2: Train a Model**. In order to build a model in a timesaving way, we used the idea of transfer learning and built our model based on VGG16. Here, we did not include top layer of VGG16 and then added a dense layer with 128 units(activation: relu), a global average pooling layer and an output layer with sigmoid activation. One thing needed to be pointed out, this architecture is chose by comparing cross entropy and accuracy with other architectures.
4. **Step 3: Predictions.** Here, we used our model to predict for our training data (tumor_091.tif) and a new tumor slide (tumor_078.tif).
5. **Step 4: Evaluations.** It seems that when the positive class is smaller and the ability to detect correctly positive samples is our main focus (detect tumor areas), it is better to use recall and precision as metrics of evaluation[1]. Thus, we chose AUC, recall, precision and F1-score as our metrics. It's because some research said that AUC is the area under the ROC curve which is commonly used as the metric to evaluate how a model performs in general. Recall (true positive rate) measures the proportion of actual positives that are correctly identified as such and precision measures number of items correctly identified as positive out of total items identified as positive. And F1-score is a harmonic mean of precision and recall given by F1=2*Precision*Recall/(Precision + Recall)[2]. However, one thing needed to be pointed out is that since our task is to detect tumors correctly and tissue regions are greatly larger than these tumor regions, we preferred a high recall under an acceptable level of precision.

2 Parameters Tuning

In this project, we came across a couple of hyper parameters that need tuning process. Therefore, we designed some experiments to decide the best value.

[1] https://towardsdatascience.com/what-metrics-should-we-use-on-imbalanced-data-set-precision-recall-roc-e2e79252aeba.

[2] https://medium.com/usf-msds/choosing-the-right-metric-for-evaluating-machine-learning-models-part-2-86d5649a5428.

2.1 Window Size, Center Size and Stride

Window size and stride decide how many patches (training data) we could get. Center size decides the labels that we should assign to each patch. In our project, we tried three combinations 299-128-128 (as the paper did), 150-100-100 and 80-50-50. Their results are shown in Fig 2-1.

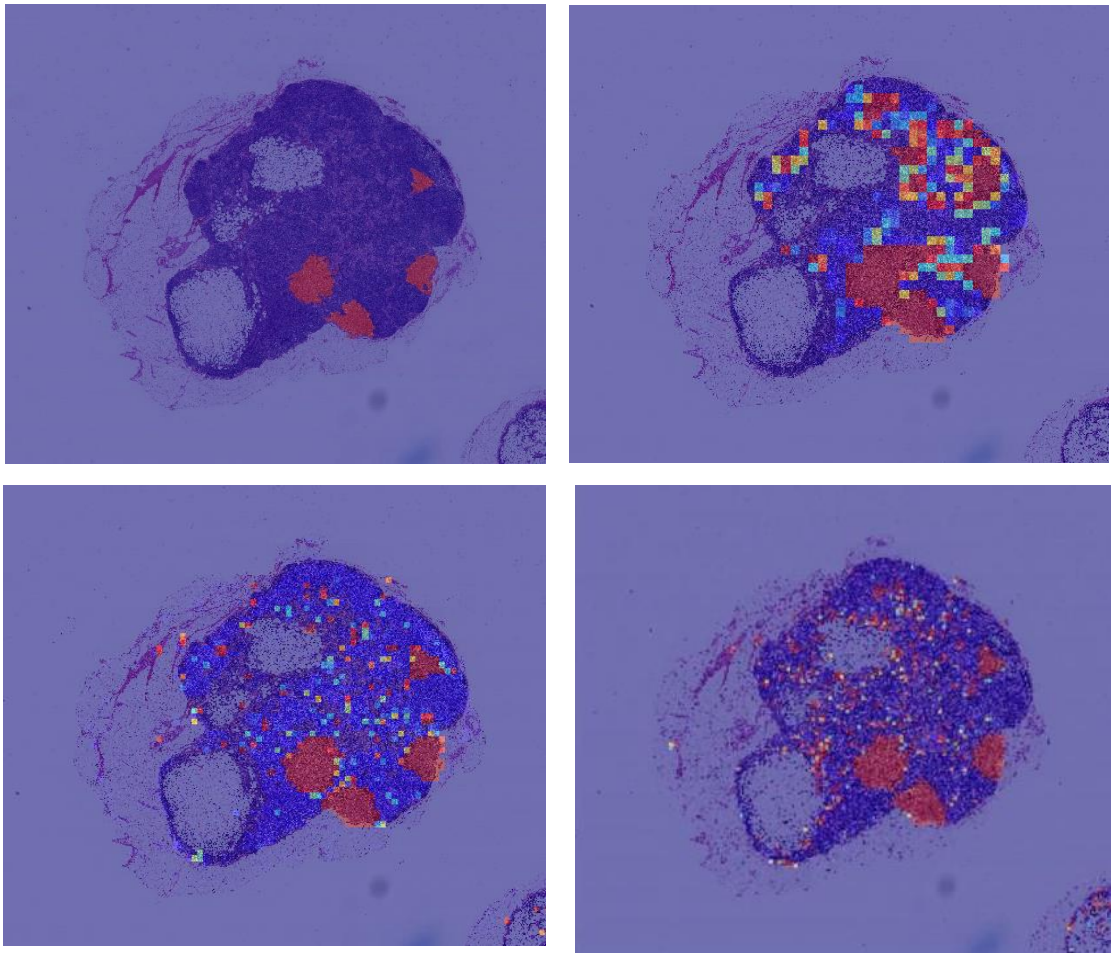


Fig 2-1: Window Size, Center Size and Stride Results (Left Top: Ground Truth, Right Top: 299-128-128, Left Bottom: 150-100-100, Right Bottom: 80-50-50)

Based on Fig 2-1, we could see that the smaller window size, center size and stride are, the more subtle result we could get. In order to get a more subtle result, we set window size to be (80, 80), center size to be (50, 50) and stride to be 50.

2.2 Data Resampling and Augmentation

As is stated before, our training data is extremely imbalanced. Therefore, we have to resample and augment our training dataset. In our project, we resampled our data in this way: if the patch is labeled as tumor, we saved it into our training dataset directly; if the

patch is labeled as normal tissue, we generated a random number from uniform distribution and when it's greater than some threshold, we would keep it. As for data augmentation, we applied 8 transformations descripted in the paper on our training dataset. During this process, threshold for resampling and if we should apply data augmentation on all training dataset have to be tuned. In our project, we tried three combinations: 0.75-Augmentation on tumor patches only, 0.75-Augmentation on all patches and 0.95-Augmentation on all patches. The results are shown in Fig 2-2.

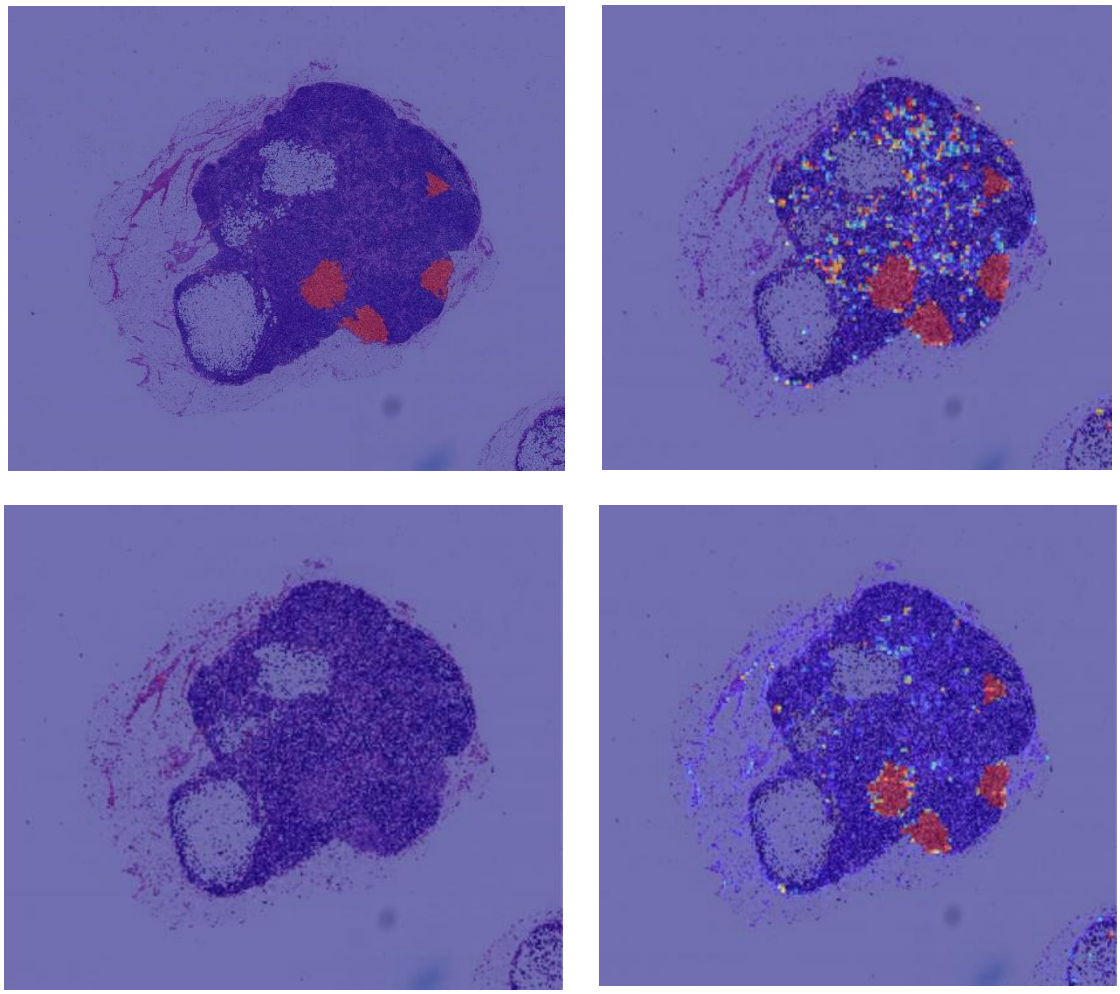


Fig 2-2: Data Resampling and Augmentation Results (Left Top: Ground Truth, Right Top: 0.75-augmentation on tumor only, Left Bottom: 0.75-augmentation on all patches, Right Bottom: 0.95-augmentation on all patches)

According to the results, we could see that when we chose 0.75 as our resampling threshold and augmented only for tumor patches, we could localize 4 main tumor regions successfully but it seemed that our model sacrificed true negative rate to do it. That is to say, 0.75-augmentation only for tumor patches could help us build a fairly good model to localize tumor areas but it would misclassify a lot of normal regions into

cancerous areas. In order to balance true negative rate and true positive rate somehow, we considered to augment all patches in our training dataset.

However, if we still chose 0.75 as our resampling threshold and augmented all patches, we got the result in Fig 2-2 Left Bottom. It is because 0.75 is not enough to make our training dataset balanced if we did data augmentations for both normal regions and tumor areas. In this case, our model would tend to classify all tumor areas into normal regions so as to get a high accuracy or low cross entropy. Thus, we increased the threshold to 0.95 and augmented all patches, we got the plot in Fig 2-2 Right Bottom. Compared with 0.75-augmentation tumor only, we think 0.95-augmentation all patches is better because it not only identified tumor regions but also it was not so “colorful” which means it did not sacrifice true negative rate so much. Therefore, it seemed that 0.95-augmentation all patches is the best choice.

In this step, we also tried to change the default binary loss function with weighted loss which means we manually increased the weights of false positive rate while training our model.

2.3 Data Preprocessings

So far, we got a model which could classify tumor regions accurately but it seemed that we could do it better. How? Let’s take a closer look at our prediction of 0.95-augmentation for all patches in Fig 2-3.

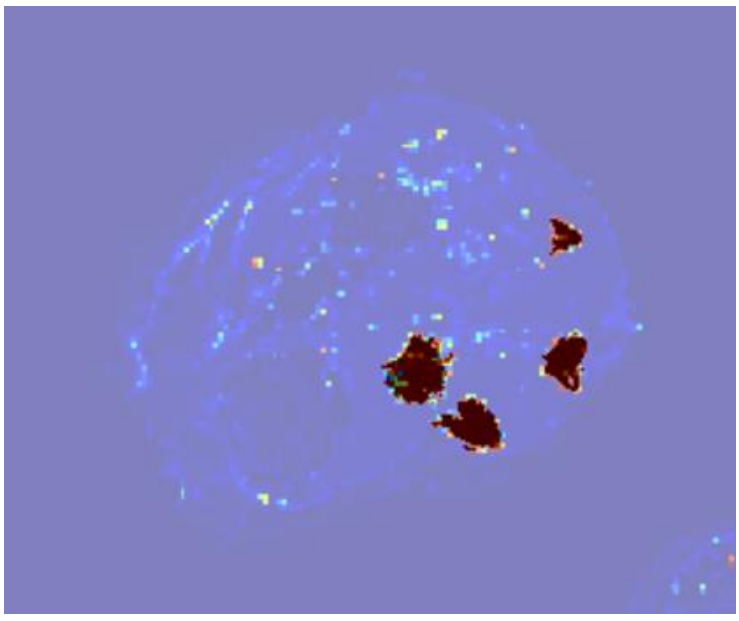
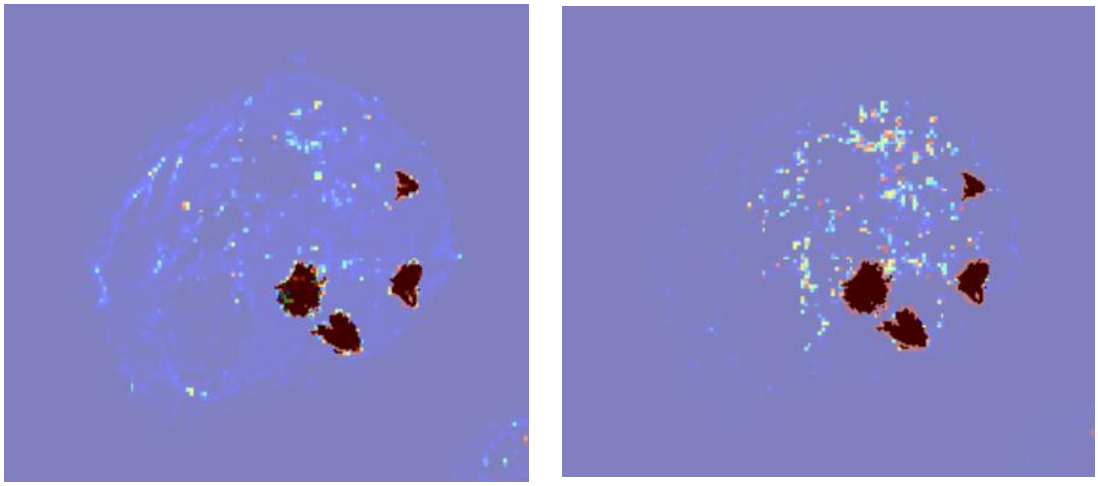


Fig 2-3: 80-50-50-0.95-Augmentation for All Patches Prediction (color blocks) with Ground Truth Mask (black)

From this plot, we could see that our model did a great job in classifying tumor with non-tumor regions but it also did the job of distinguished tissue areas from non-tissue areas. However, it seemed that we do not have to do that. So in order to improve



efficiency and focus on localizing tumor regions only, we also tried to perform foreground segmentation before training a model. The results are shown in Fig 2-4.

Fig 2-4: Left: Prediction without Foreground Preprocessing

Right: Prediction with Foreground Preprocessing

After preprocessing foreground, we could see that our model did not take the task of classifying tissue areas and normal regions anymore. It is hard to tell which one is better just based on plots above. Therefore, we also randomly pick a test slide (*tumor_078.tif*) to evaluate our model in general. The test results are shown in Fig 2-5.

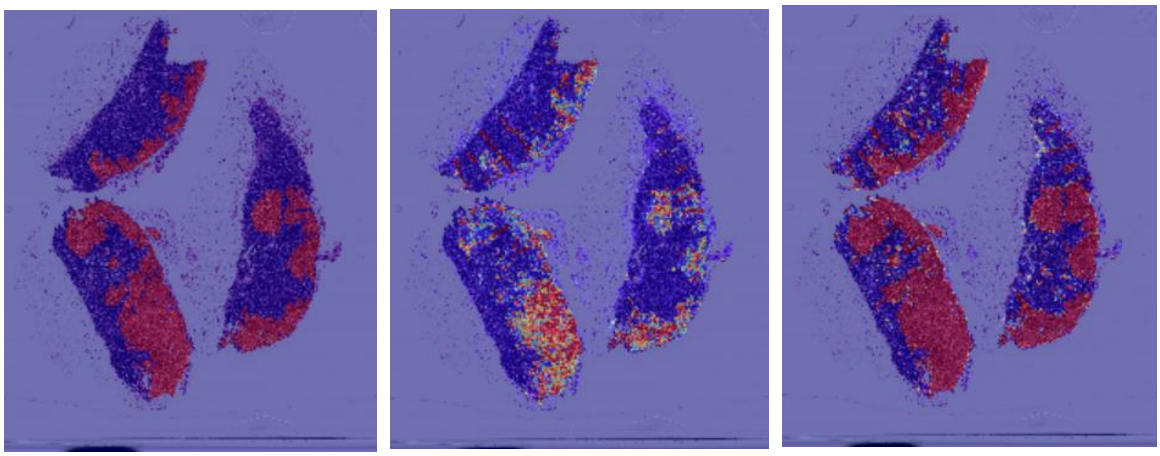


Fig 2-5: Left to Right: Ground Truth, Prediction without finding tissues and Prediction with finding tissues

From Fig 2-5, we could see that if we do not preprocess foreground, it seemed that our model over-fitted training data a little bit so that for a new test slide, our model could not perform as well as training data. However, if we preprocess foreground before

building a model, our model could still do a great job on a new test slide. Therefore, it seemed that it's worthy to do foreground preprocess before training a model which could help us improve the model's ability of generalization.

2.4 Multi-Scale Model

Besides tuning on single-scale model, we also tried simple multi-scale model. Parameters for our multi-scale model are shown in Table 2-1.

Table 2-1: Parameters for Multi-Scale Model

Parameters	Values
Window Size	(300, 300)
Center Size	(150, 150)
Stride	128
Resampling Threshold	0.95
Data Augmentation	Augment tumor
Foreground Preprocessing	Yes

First, we used two input into two sub-models to preprocess input data. After several Convolutional layers and MaxPooling layers, we merged the two sub-models into our combined model. The architecture of this multi-scale model is shown as below.

Table 2-2: Architecture of multi-scale model

Parameters	Sub-model 1	Sub-model 2
Input Layer	(300, 300, 3)	(150, 150, 3)
Layer 1	Conv2D (32, 3, 3)	Conv2D (32, 3, 3)
Layer 2	MaxPooling2D (2, 2)	MaxPooling2D (2, 2)
Layer 3	Conv2D (64, 3, 3)	Conv2D (64, 3, 3)
Layer 4	MaxPooling2D (2, 2)	
Current Layer Size	(64, 75, 75)	(64, 75, 75)
Multi-Layer 1	MaxPooling2D (2, 2)	
Multi-Layer 2	Dense (128)	
Multi-Layer 2	Dense (1)	
Activation Function	Sigmoid	

We used RMSProp Optimizer with learning rate 0.001 to train our combined model with 10 epochs and 10 batch size. Our result for training slide (*tumor_091.tif*) is a

follows.

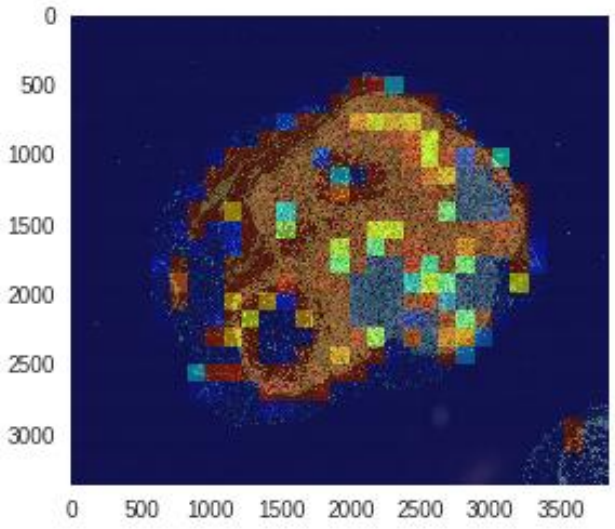


Fig 2-6: Result of Training Slide

According to this plot, we could say that our multi-scale model does not perform well, at least it cannot outperform the single-scale model that we built above. Therefore, we chose single-scale model as our final model.

3 Final Model and Results

3.1 Details of Final Model

Based on our tuning process, the parameters for our final model are shown in Table 3-1.

Table 3-1: Parameters for Our Final Model

Parameters	Values
Window Size	(80, 80)
Center Size	(50, 50)
Stride	50
Resampling Threshold	0.95
Data Augmentation	Augment All
Foreground Preprocessing	Yes

We used RMSProp Optimizer with learning rate 0.0001 to train our model with 8

epochs and 32 batch size. Our results for training slide (*tumor_091.tif*) and a tumor test slide (*tumor_078.tif*) are as follows.

3.2 Results for Training Slide

Our prediction for training slide is shown in Fig 3-1.

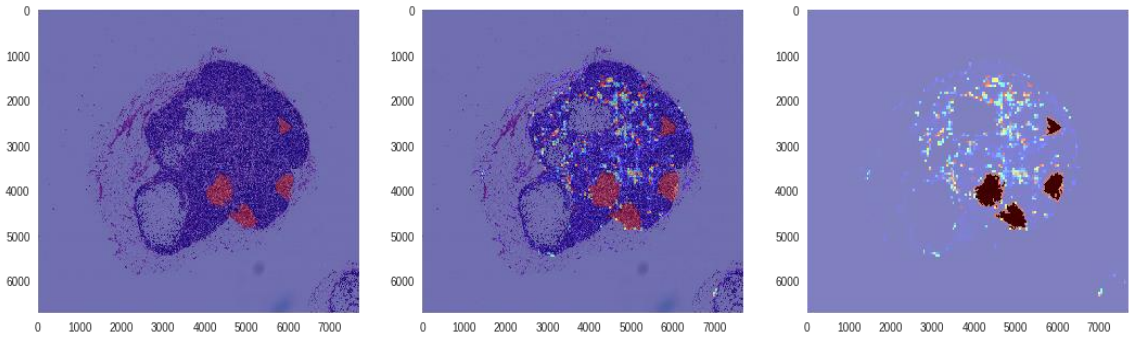


Fig 3-1: Results of Training Slide

And for this slide, the confusion matrix is shown in Table 3-2. And AUC is 0.99795, recall is 0.99486, precision is 0.5545 and F1-score is 0.72. As stated before, since our main task is to localize tumor successfully, we preferred a high recall under an acceptable level of precision. In this case, we could think that our model performs well for our goal.

Table 3-2: Confusion Matrix for Training Slide

	Prediction: Non-tumor	Prediction: Tumor
Truth: Non-tumor	50263252	598136
Truth: Tumor	3848	744364

3.3 Results for New Test Slide

Our prediction for training slide is shown in Fig 3-2.

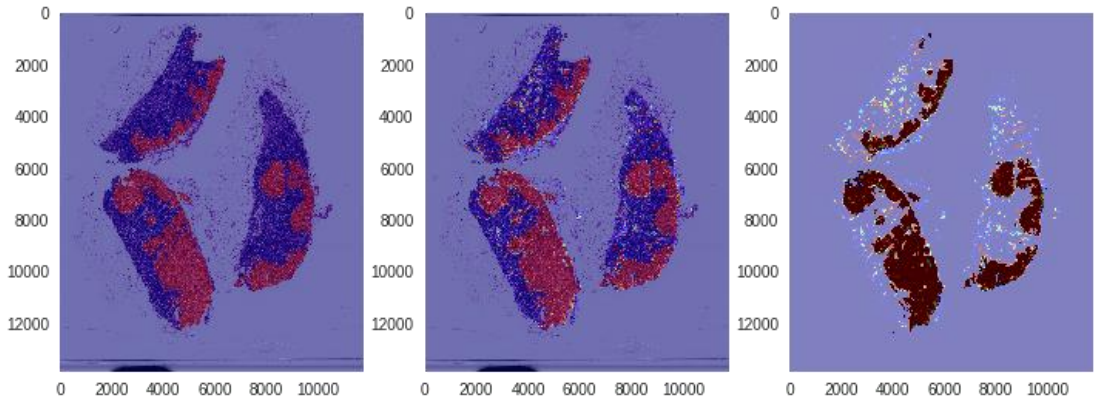


Fig 3-2: Results of Test Slide

And for this slide, the confusion matrix is shown in Table 3-3. And AUC is 0.98444, recall is 0.97334, precision is 0.73139 and F1-score is 0.84. It seems that our model got a good prediction for this new test slide.

Table 3-3: Confusion Matrix for Test Slide

	Prediction: Non-tumor	Prediction: Tumor
Truth: Non-tumor	142950556	5423240
Truth: Tumor	404532	14766760

4 Summary

In this project, we used the idea of transfer learning to detect tumor on gigapixel pathology images. Our final model performs well both on training slide and a random test slide. What's more, in the process of tuning, we also found that: (1) the smaller window size, center size and stride are, the more subtle result we could get; (2) resampling and data augmentation play an important role for detection when our dataset is extremely imbalanced. In our task, augmenting all data is better than augmenting tumor patches only because it did not sacrifice true negative rate so much; (4) it's worthy to do foreground preprocess before training a model which could help us improve the model's ability of generalization; (5) multi-scale model cannot outperform single-scale model because it would take a long time to train and its results are not subtle.

In the future work, we hope to speed up our model and it seems that we could somehow extend our model so as to detect tumor real-time. For example, in neural style transfer topic, we added a transform net to increase the speed of prediction. Perhaps, we could try to introduce this idea to our tumor detection task so as to do it in real-time.