Detecting Cancer Metastases on Gigapixel Pathology Images - Applied Deep Learning Final Project

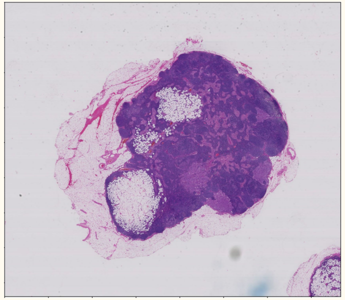
Yanru Chen (yc4037@columbia.edu)

Introduction

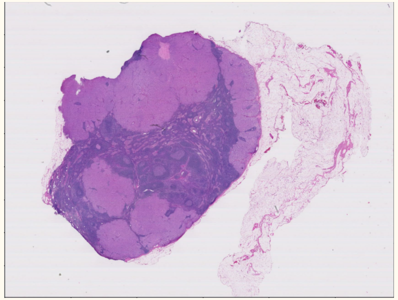
- Recreate some of data pre-processing, data augmentation, model training, inference and results generation from paper “Detecting Cancer Metastases on Gigapixel Pathology Images”
- Goal: Create an end-to-end model that takes cancer slides as input and generate predicted cancer heatmap as output
- Part of the paper included in the project:
    - Including multiple slides in training set
    - Including multiple zoom levels in training set
    - Used flip & rotate to augment tumor images
    - Used data augmentation and image normalization
    - Trained InceptionV3 and compared results from transfer learning and training from scratch
    - Use different evaluation metrics

Data Overview

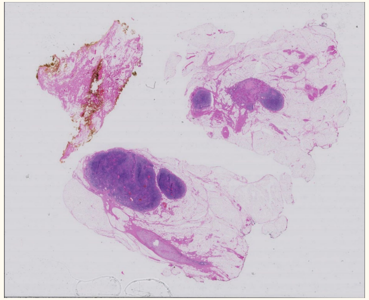
- Use 3 slides of training data from CAMELYON16:
  - Training: 91, 110
  - Testing: 91, 110 (internal testing) , 94 (external testing)
- Slide 91: lots of non cancerous cell, four small patch of cancerous cell
- Slide 110: large portion of cancerous cell
- Slide 94: one small patch of cancerous cell, used for model's external testing purposes



91



110



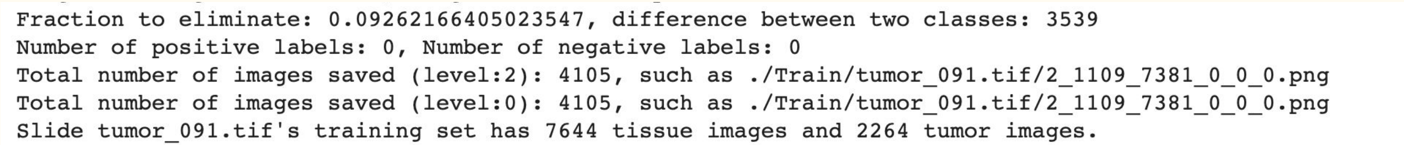
94

Data Preprocessing

- Steps to create training dataset:
    - Step 1: Model use a random patch extraction window of size 128*128 to generate labels and window of size 299*299 to create patches
        - Slide 91’s tumor patch will be augmented using rotate and flip techniques
        - Tumor patches will be undersampled if data is imbalanced
    - Step 2: all patches are saved into folder separated by slide name and with image name format of zoom_level, x coordinate, y coordinate, rotation count, flipped, is_tumor
        - Each variable separated by “_”
    - Step 3: after all patches saved as png into their designated folders in path “/content/Train/”, the program will read all training images into different data frames, separated by zoom levels
    - Step 4: data frames of different zoom levels will be joined
        - Dataframe merge key: slide name, x, y, rotation count, flipped, is_tumor
        - Different zoom level slides will be paired for Tensorflow dataset creation
    - Step 5: Dataset will receive (image path 1, image path 2, has tumor) pairs for loading the images
    - Step 6: all images will be loaded, decoded, resized, normalized, creating the final training dataset

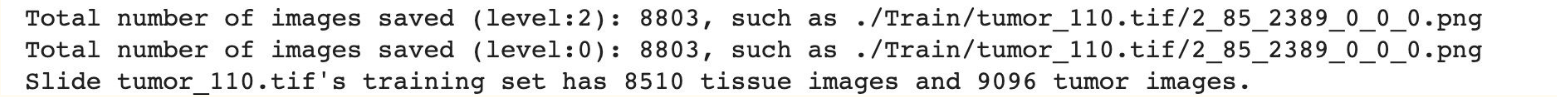
Training Set w/ zoom level 2 & 0

- For slide 91, 9% of non tumor patches are removed from training set where the final difference between augmented tumor slides and undersampled tumor slides are 3539
  - Since slide 110 has more tumor patches, we don't need to augment more tumor patches from slide 91
  - We have 7644 normal tissue patches and 2264 tumor patches in total



```
Fraction to eliminate: 0.09262166405023547, difference between two classes: 3539
Number of positive labels: 0, Number of negative labels: 0
Total number of images saved (level:2): 4105, such as ./Train/tumor_091.tif/2_1109_7381_0_0_0.png
Total number of images saved (level:0): 4105, such as ./Train/tumor_091.tif/2_1109_7381_0_0_0.png
Slide tumor_091.tif's training set has 7644 tissue images and 2264 tumor images.
```

- For slide 110, no undersampling is performed, since the patches are mostly in balance
  - We have total of 8510 normal tissue images and 9096 tumor images, both are not augmented and undersampled



```
Total number of images saved (level:2): 8803, such as ./Train/tumor_110.tif/2_85_2389_0_0_0.png
Total number of images saved (level:0): 8803, such as ./Train/tumor_110.tif/2_85_2389_0_0_0.png
Slide tumor_110.tif's training set has 8510 tissue images and 9096 tumor images.
```

Data Overview

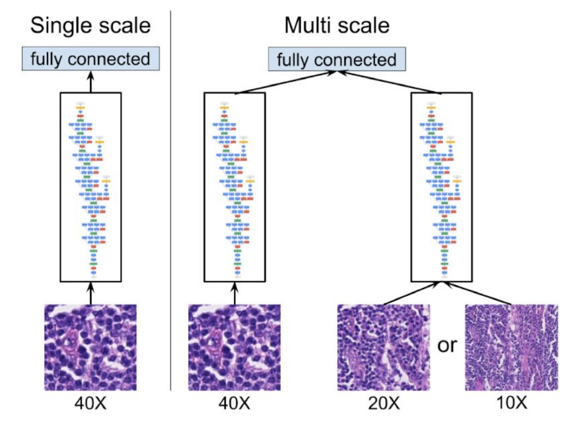
- Zoom level combination:
    - Two zoom level are chosen (trained and tested)
        - Model 2: (3,1)
        - Model 3: (2,0)
    - Model 1: only zoom level 2 are chosen, model 1 only uses 1 zoom level (since it is the first baseline model developed)
- Data augmentation:
    - Built as a layer of tensorflow model
    - Used the same data augmentation process in paper
        - Brightness with max delta of 64/255
        - Saturation with max delta of 0.25
        - Hue with a max delta of 0.04
        - Contrast with max delta of 0.75
- Lastly, batched training set was created with batch size 8

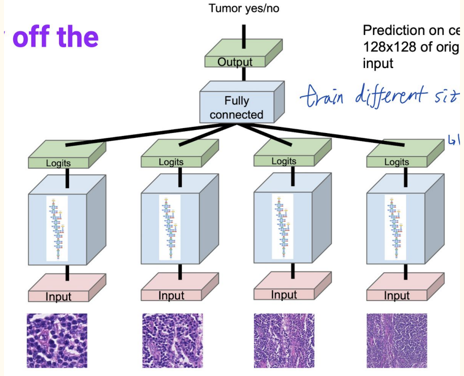
Testing Set

- Testing set are used for testing and inference purposes
- Pipeline:
  - For each pixel at coordinate (x, y) at slide of zoom level 7 (480*420), the program will get the corresponding patch at zoom level A and zoom level B, where A & B need to **less than 7** and **equal to the level used for training**
    - Both combination of (3,1) and (2,0) are tested
    - (x, y) pixel will be the center of the generated patch
  - Since 480*420 are still too much pixels and takes way too long to run, upsampling factor z was used where only center pixel of z*z matrix was used for prediction
    - All pixels inside that z*z matrix will be broadcasted with predicted center label
    - z = 5 was used in model 2 and model 3, z = 3 was used in model 1 (baseline)
  - Background patch with < 20% tissue pixels will be automatically assigned label 0 (normal tissue label)
- Each extracted patch will be created as a Tensorflow dataset with batch size of 1 for easier predicting, normalization will be applied
- The final dataset will be of format (patch from zoom level A, patch from zoom level B), placeholder label

Model

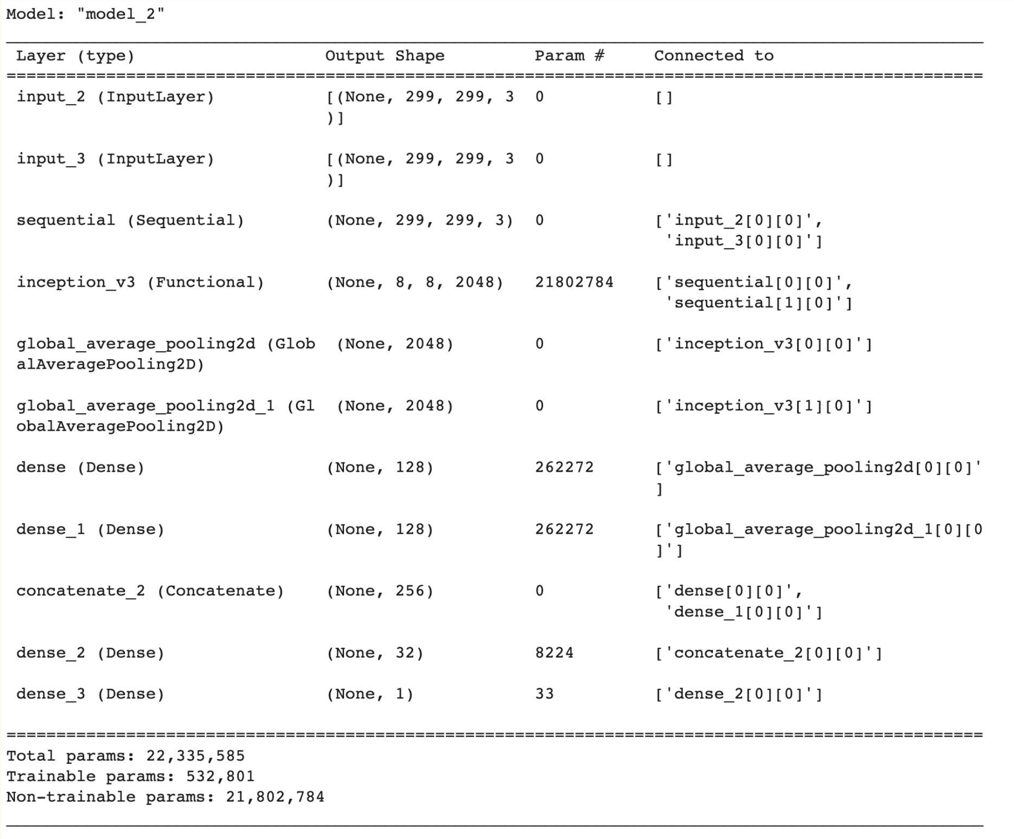
- Model structure introduced in paper are used:
    - Model Type:
        - Single scale:
            - Pre-trained InceptionV3 on zoom level 2 with only slide 91
        - Multi scale:
            - Pre-trained InceptionV3 on ImageNet on zoom level (3,1), uses slide 91 and 110
            - InceptionV3 with trainable weights on zoom level (2,0), uses slide 91 and 110
            - Encoded vectors from both zoom level models are concatenated and putted into a dense layer of size 32, with relu activation function
            - The final output layer used sigmoid as its activation function





Model Structure

- Optimizer:
  - Same optimizer RMSProp with momentum of 0.9, decay of 0.9, epsilon of 1 was used
  - Starting learning rate is 0.002 for all models
- Loss function:
  - BinaryCrossentropy was used
- Model evaluation metric during training:
  - Binary accuracy
- Model 3 was trained using 10 epochs and model 2 was trained using 2 epochs



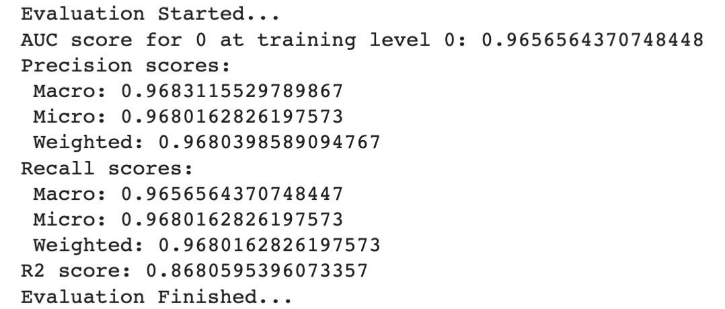
```
Model: "model_2"
__________________________________________________________________________________________________
 Layer (type)                   Output Shape         Param #     Connected to
==================================================================================================
 input_2 (InputLayer)           [(None, 299, 299, 3  0           []
                                )]

 input_3 (InputLayer)           [(None, 299, 299, 3  0           []
                                )]

 sequential (Sequential)        (None, 299, 299, 3)  0           ['input_2[0][0]',
                                                                  'input_3[0][0]']

 inception_v3 (Functional)      (None, 8, 8, 2048)   21802784    ['sequential[0][0]',
                                                                  'sequential[1][0]']

 global_average_pooling2d (Glob  (None, 2048)        0           ['inception_v3[0][0]']
 alAveragePooling2D)

 global_average_pooling2d_1 (Gl  (None, 2048)        0           ['inception_v3[1][0]']
 obalAveragePooling2D)

 dense (Dense)                  (None, 128)          262272      ['global_average_pooling2d[0][0]'
                                                                 ]

 dense_1 (Dense)                (None, 128)          262272      ['global_average_pooling2d_1[0][0
                                                                 ]']

 concatenate_2 (Concatenate)    (None, 256)          0           ['dense[0][0]',
                                                                  'dense_1[0][0]']

 dense_2 (Dense)                (None, 32)           8224        ['concatenate_2[0][0]']

 dense_3 (Dense)                (None, 1)            33          ['dense_2[0][0]']

==================================================================================================
Total params: 22,335,585
Trainable params: 532,801
Non-trainable params: 21,802,784
__________________________________________________________________________________________________
```

Evaluation Metrics

- Scikit-learn library was used to create testing results for the models
- 4 metrics will be computed in addition to binary accuracy produced by Tensorflow (all the metrics compute slide-level classification accuracy: thresholded model predictions compared to the ground truth):
  - AUC:
    - This was used as part of the evaluation metric in the original paper, which tells the model's ability to classify observations between positive and negative classes
  - Precision (macro, micro, weighted)
    - Compute the number of correct tumor predictions made
    - Used because I want to know how good the model is correctly labeling tumor cells, and not flagging non tumor cell as tumor
  - Recall:
    - Compute how many, of all tumor cells, can model correctly predict
    - Used because I want to know what percentage of tumor can model actually predict
  - R2 score:
    - Compute the tradeoff between tumor cell
    - Used because I want to know what is the tradeoff between precision and recall

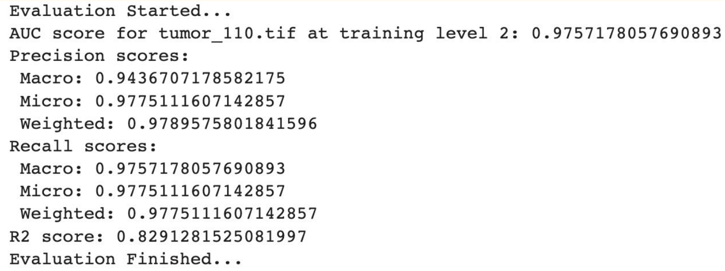
Results (Model 3)

Evaluation on training dataset (slide 91, 110):

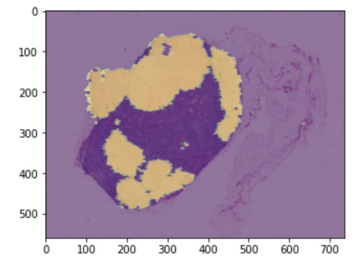


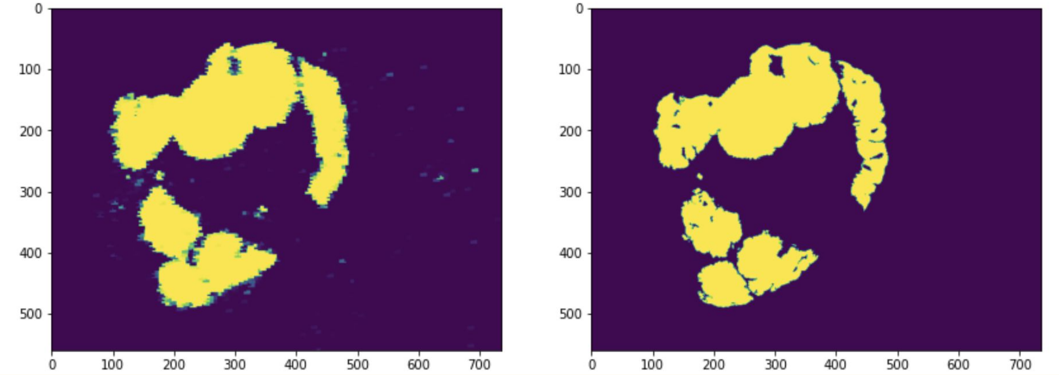
```
Evaluation Started...
AUC score for 0 at training level 0: 0.9656564370748448
Precision scores:
 Macro: 0.9683115529789867
 Micro: 0.9680162826197573
 Weighted: 0.9680398589094767
Recall scores:
 Macro: 0.9656564370748447
 Micro: 0.9680162826197573
 Weighted: 0.9680162826197573
R2 score: 0.8680595396073357
Evaluation Finished...
```

Internal testing (slide 110):



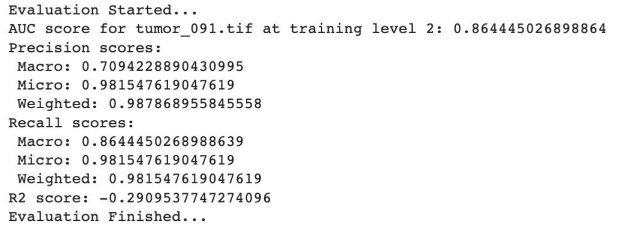
```
Evaluation Started...
AUC score for tumor_110.tif at training level 2: 0.9757178057690893
Precision scores:
 Macro: 0.9436707178582175
 Micro: 0.9775111607142857
 Weighted: 0.9789575801841596
Recall scores:
 Macro: 0.9757178057690893
 Micro: 0.9775111607142857
 Weighted: 0.9775111607142857
R2 score: 0.8291281525081997
Evaluation Finished...
```



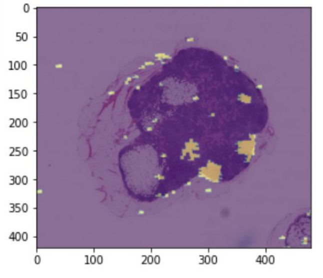


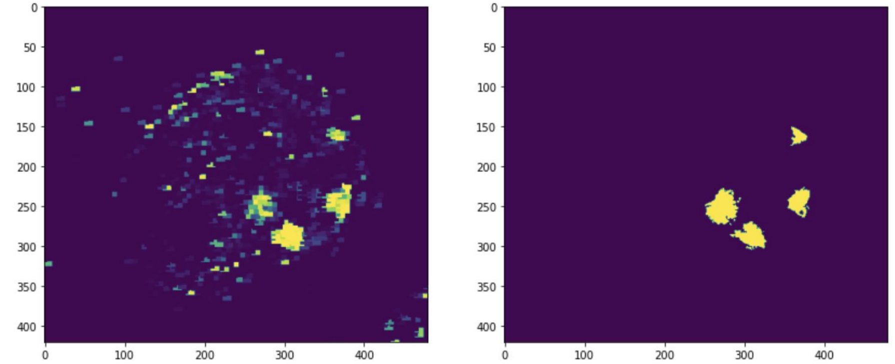
Results (Model 3)

Internal testing for slide 91:

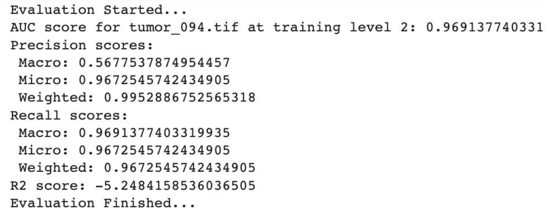


```
Evaluation Started...
AUC score for tumor_091.tif at training level 2: 0.864445026898864
Precision scores:
 Macro: 0.7094228890430995
 Micro: 0.981547619047619
 Weighted: 0.987868955845558
Recall scores:
 Macro: 0.8644450268988639
 Micro: 0.981547619047619
 Weighted: 0.981547619047619
R2 score: -0.2909537747274096
Evaluation Finished...
```

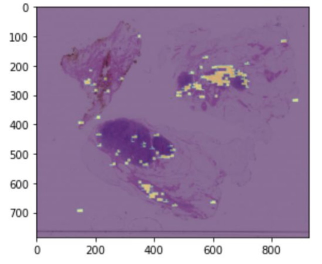


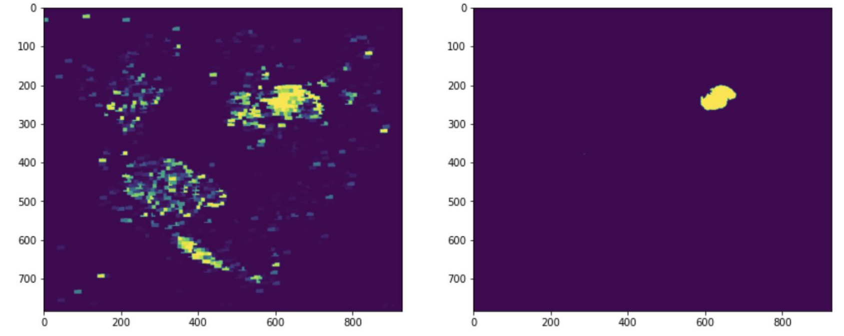


External testing for slide 94:

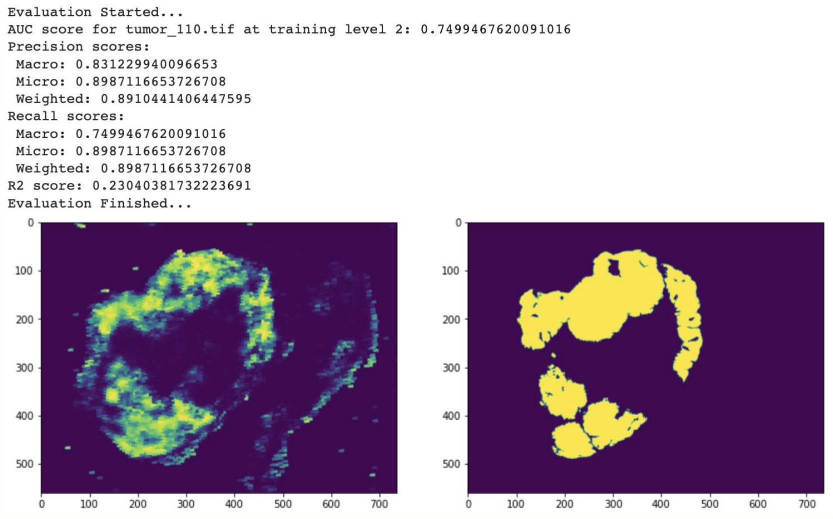


```
Evaluation Started...
AUC score for tumor_094.tif at training level 2: 0.969137740331
Precision scores:
 Macro: 0.5677537874954457
 Micro: 0.9672545742434905
 Weighted: 0.9952886752565318
Recall scores:
 Macro: 0.9691377403319935
 Micro: 0.9672545742434905
 Weighted: 0.9672545742434905
R2 score: -5.2484158536036505
Evaluation Finished...
```

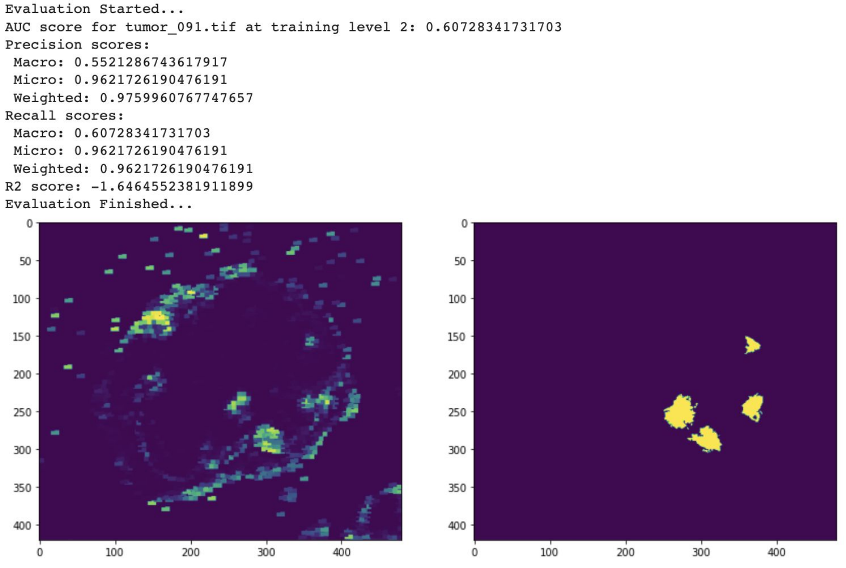




Results (Model 2)

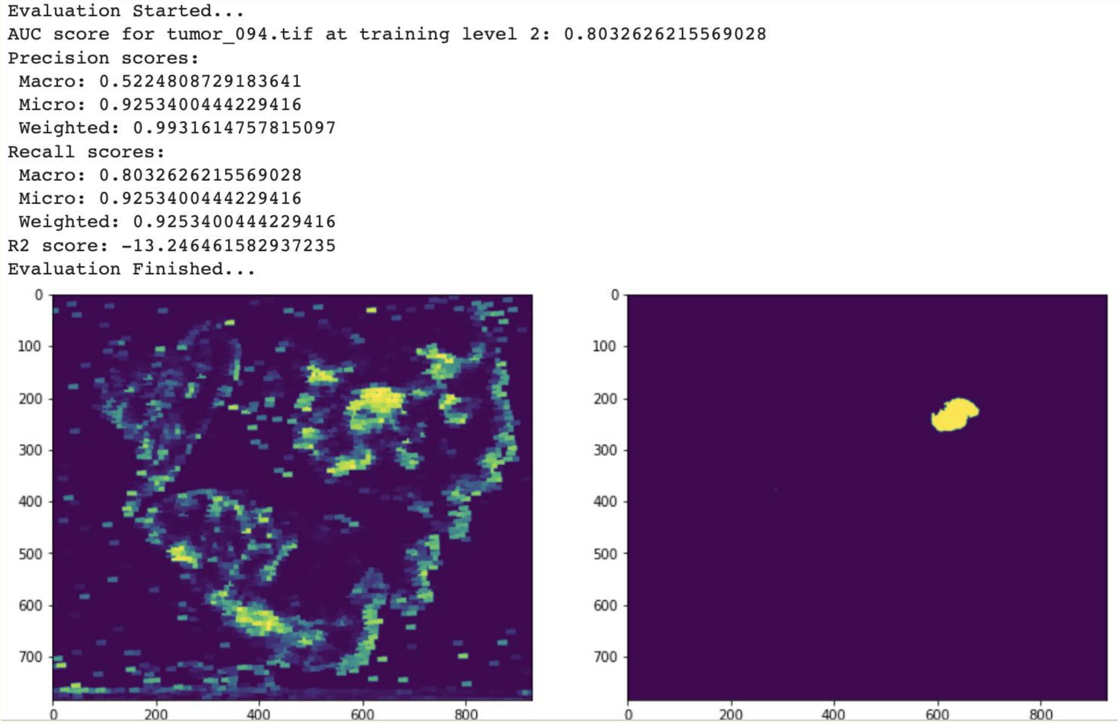


```
Evaluation Started...
AUC score for tumor_110.tif at training level 2: 0.7499467620091016
Precision scores:
 Macro: 0.831229940096653
 Micro: 0.8987116653726708
 Weighted: 0.8910441406447595
Recall scores:
 Macro: 0.7499467620091016
 Micro: 0.8987116653726708
 Weighted: 0.8987116653726708
R2 score: 0.23040381732223691
Evaluation Finished...
```



```
Evaluation Started...
AUC score for tumor_091.tif at training level 2: 0.60728341731703
Precision scores:
 Macro: 0.5521286743617917
 Micro: 0.9621726190476191
 Weighted: 0.9759960767747657
Recall scores:
 Macro: 0.60728341731703
 Micro: 0.9621726190476191
 Weighted: 0.9621726190476191
R2 score: -1.6464552381911899
Evaluation Finished...
```

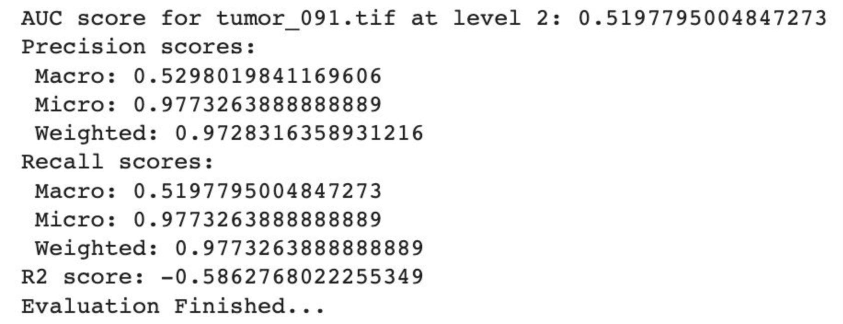
Results (Model 2): external testing



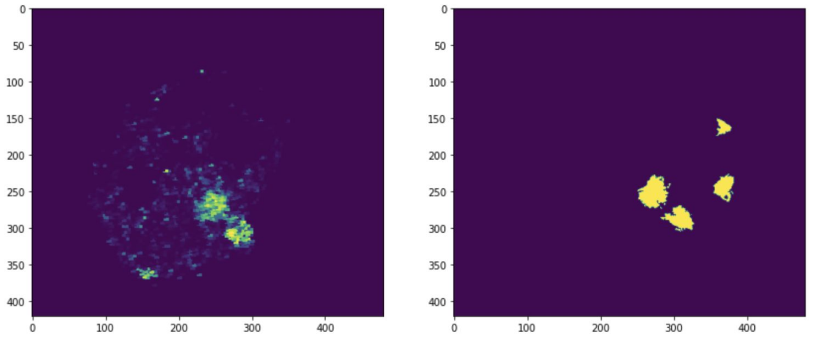
```
Evaluation Started...
AUC score for tumor_094.tif at training level 2: 0.8032626215569028
Precision scores:
 Macro: 0.5224808729183641
 Micro: 0.9253400444229416
 Weighted: 0.9931614757815097
Recall scores:
 Macro: 0.8032626215569028
 Micro: 0.9253400444229416
 Weighted: 0.9253400444229416
R2 score: -13.246461582937235
Evaluation Finished...
```

Results (Model 1):

Internal testing for model 1:



```
AUC score for tumor_091.tif at level 2: 0.5197795004847273
Precision scores:
 Macro: 0.5298019841169606
 Micro: 0.9773263888888889
 Weighted: 0.9728316358931216
Recall scores:
 Macro: 0.5197795004847273
 Micro: 0.9773263888888889
 Weighted: 0.9773263888888889
R2 score: -0.5862768022255349
Evaluation Finished...
```



Analysis

- Model 1, which used the least amount of data with only 1 zoom level (2), performed the worst with AUC only ~52
- Model 2, which has lower zoom level combination (3 & 1), has higher AUC than model 1 on internal testing set: 75 on slide 110 and 60 on slide 91
  - AUC of 52 on external testing set, slide 94
- Model 3, which has higher zoom level combination (2&0) and trainable weights, has highest AUC for both internal and external testing set
  - Slide 110: AUC 94
  - Slide 91: AUC 71
  - Slide 94: AUC 57
- The same pattern can also be observed in precision and recall
- We can conclude that higher zoom level, more training data, and non-transfer learning model helps with improving model performance