

# Tumor Detection with Deep Learning

Columbia University - Applied Deep Learning Project

Jinwoo Jung (jj2762@columbia.edu), Hyuk Joon Kwon (hk3084@columbia.edu)

Github repo: [https://github.com/bluecube246/Tumor\\_Semantic\\_Image\\_Segmentation](https://github.com/bluecube246/Tumor_Semantic_Image_Segmentation)

Readme: [https://github.com/bluecube246/Tumor\\_Semantic\\_Image\\_Segmentation/blob/master/README.md](https://github.com/bluecube246/Tumor_Semantic_Image_Segmentation/blob/master/README.md)

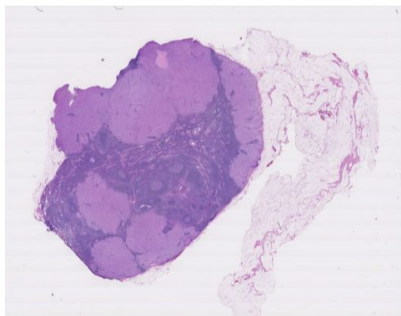


# Layout

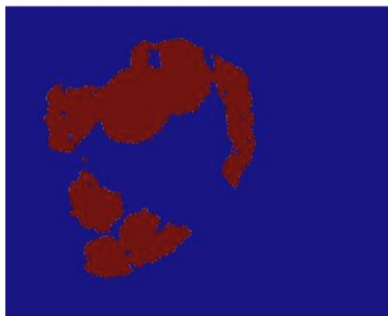
1. Project Overview and Objective
2. Initial Data Visualization
3. Methodology
  - a. Sliding window approach
  - b. Minimum tissue percentage
  - c. Balancing data
  - d. Multiple zoom levels and labeling sub-patch
4. Models
  - a. Basic model: 2 InceptionV3 models
    - i. 4 variations to the basic model
  - b. Fancier model:
    - i. 3 variations to the fancier model
5. Predictions
  - a. Using zoom level 3,4,5    stride 150
  - b. Using zoom level 2,3    stride 200
  - c. Using zoom level 2,3,4    stride 299
  - d. Using zoom level 2,3,4    stride 200
  - e. Using zoom level 1,2    stride 299
  - f. Using zoom level 0,1    stride 299
6. Conclusion

# Project Overview

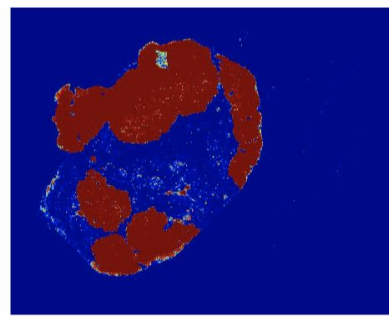
This project is to build a tool that produces heatmaps that identify regions of tumors in tissues of biopsy images that works as an automatic second opinion for pathologists. The provided sample datasets are originally from CAMELYON16 Challenge's 400 WSI( whole slide images), which are collected independently from two medical centers in the Netherlands.



**Biopsy image**



**Ground truth  
(from pathologist)**



**Model predictions**



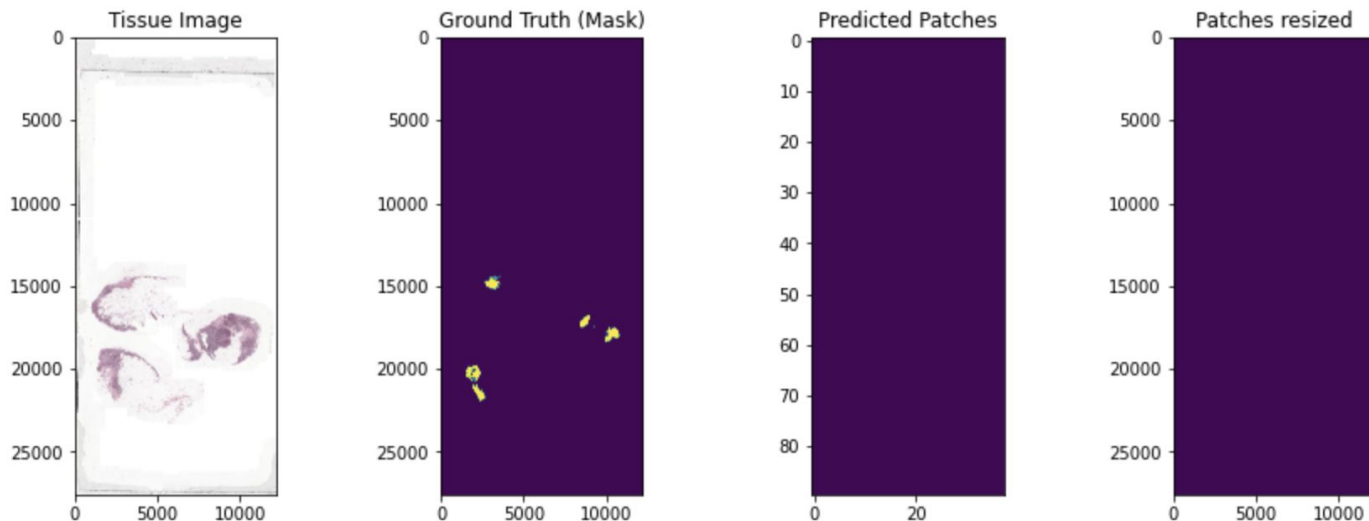
# Objective

- It is important to decide on what makes a model good.
- Due to the extreme imbalance of Tumor and Normal patches such that there are much more patch images that are classified as having tumor than not, a lot of the images will achieve over a 99 percent accuracy by simply predicting all the patches to be Normal. (Example of this issue shown next slide)
- It would more be problematic if a model predicted a patch to be normal when it actually contained cancer.
- Goal: our objective is to try and achieve a high true positive rate (equally low false negative rate) at the same time trying to achieve high AUC accuracy.

# High accuracy but bad model example.

Below is an example of a model that has a high accuracy by simply predicting all patches as 0 (Normal)

```
107/107 [=====] - 21s 122ms/step - loss: 0.0415 - accuracy: 0.9926  
12224 27584
```





## 2. Initial Data Visualization

- The dataset is composed of 21 biopsy and respective mask images.
  - (img number 38 was taken out for consideration, due to difference in level dimensions)
- Decided to not use 6 of the images that rarely contain any tumors.
- We further selected images that goes into train/val/test set such that 'good' biopsy and mask images are distributed more toward training set and test set.
- Details on the initial visualization can be found in the following file:  
*0.Initial\_Data\_Visualization.ipynb* under the Github repo

Biopsy/mask img number in

- train set: [5, 16, 19, 23, 31, 84, 94, 101, 110]
- valid set: [1, 75, 96]
- test set: [64, 78, 91]



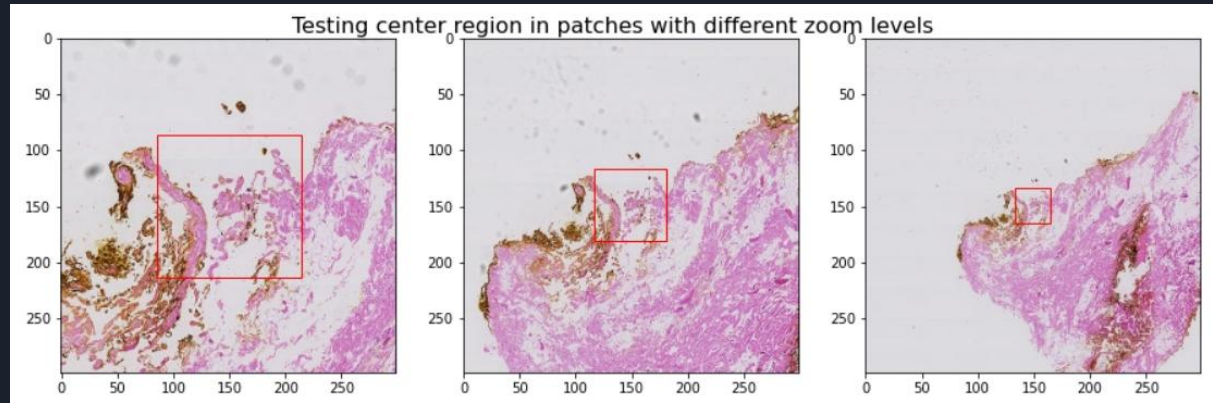
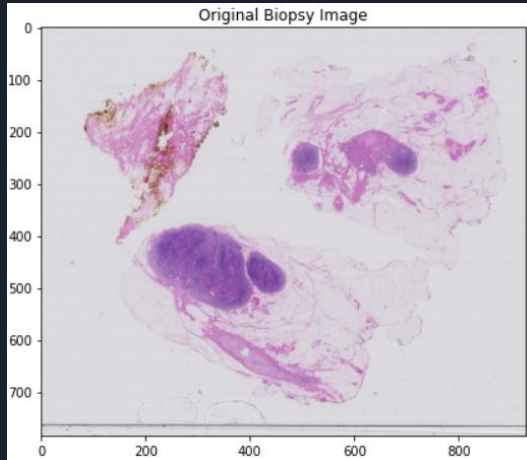
### 3. Methodology

- a. Sliding Window approach
  - Method used in creating patches for each biopsy and mask slides
  - Creates a list of patches from top to bottom, from left to right
  - Slides over each patch with predefined stride size (ex. 150, 200, 299)
  
- b. Minimum tissue percentage
  - Stores only patches that have tissue pixel percentage above 50%
  - Helps to reduce size of data need to store in RAM
  
- c. Balancing data
  - To overcome imbalance number of patches that have tumors and those without
  - Undersampled patches of the majority class

### 3. Methodology

#### d. Multiple zoom levels and classifying sub-patches

- Transformed coordinates to locate same position throughout multiple zoom levels
- Performed sliding in coordinates with reference to the lowest level to prevent highest level patch move outside to the available coordinates.
- Labeled/classified each patch by looking at center sub-patch region of 128x128, labeled 1 for having tumor if at least a single pixel in the sub-patch has a tumor.





## 4. Models

- a. Basic model: 2 InceptionV3 models + dense(1024) + dense(1)  
Version a: basic mdl (w/o fine tuning, w/o data augmentation)  
Version b: basic mdl w/ fine tuning, w/o data augmentation  
Version c: basic mdl w/o fine tuning, w/ data augmentation  
Version d: basic mdl w/ fine tuning, w/ data augmentation

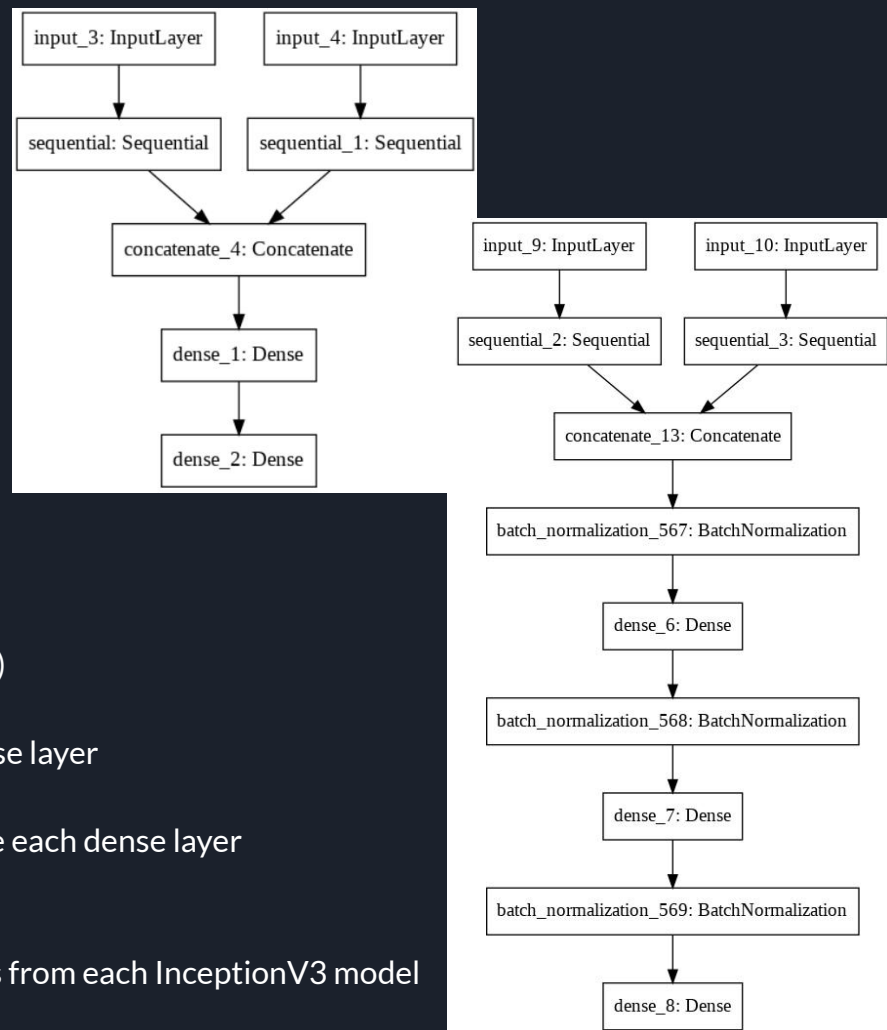
- b. Fancier model for version 2.x:  
2 InceptionV3 models + dense (1024) + dense (128) + dense (1)

Version 2.a: fancier mdl + Batch Normalization before each dense layer

Version 2.b: fancier mdl + Dropout before each dense layer

Version 2.c: fancier mdl + Batch Normalization + Dropout before each dense layer

Note: fine tuning -> fine tuned starting at 100th layer out of 311 layers from each InceptionV3 model





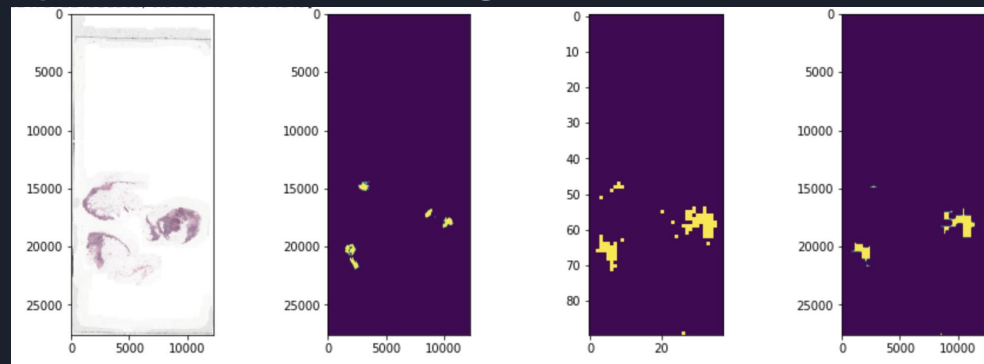
## 5.a Predictions (zoom level 3,4,5 & stride 150)

- No data balancing was done.
- Overall the model predicted very poorly for all three images
- The model predicted most patches to be Normal. (Expected due to the absence of data balancing.)
- As a result Image 78 has a much lower score because it had more patches that were actually Tumor

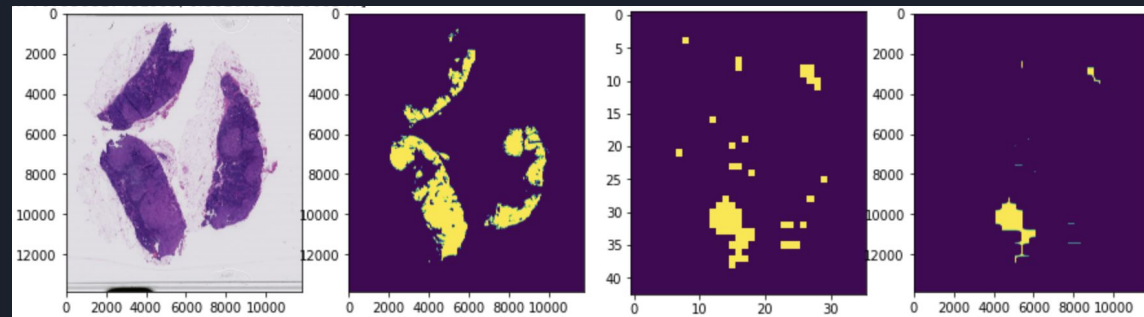
Image number	64	78	91
Best model	Version a	Version a	Version b
False Negative Rate	0.99	0.87	0.39
True Positive Rate	0.62	0.13	0.61
AUC	0.8	0.56	0.81

# Best Predictions from previous setting

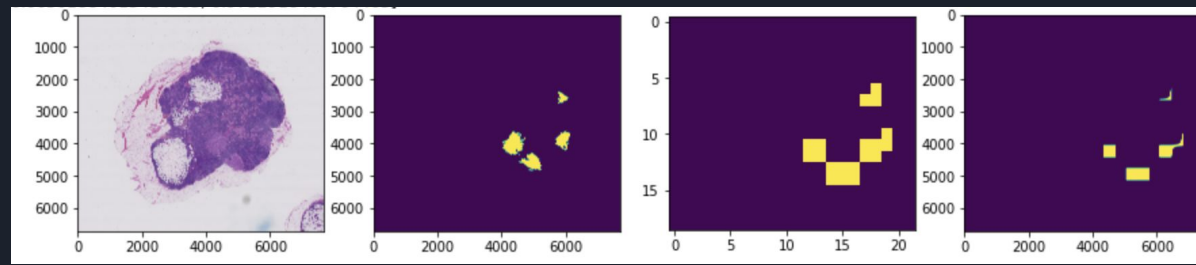
Version a's prediction on img 64:



Version a's  
prediction on img 78:



Version b's  
Prediction on img 91:





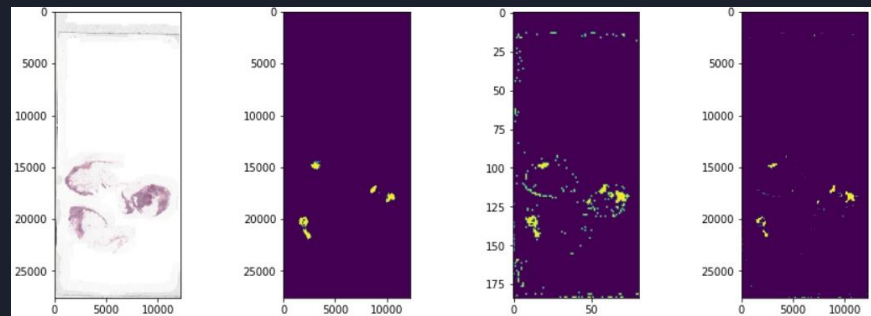
## 5.b Predictions (Base Using zoom level 2,3 & stride 200)

- Adopted larger stride of 200 as we decreased zoom level
- There does not exist a single model that outperforms other models in predictions
- Simply highest accuracy is not a holistic measure to decide a best model
  - Exists a model that has lower accuracy but produces better heatmaps
    - Over predicting tumor labels (ex. Version 2.a's prediction on img num 91)

Image number	64	78	91
Best model	Version c	Version 2.a	Version 2.a
False Negative Rate	0.35	0.24	0.2
True Positive Rate	0.65	0.76	0.8
AUC	0.82	0.86	0.86

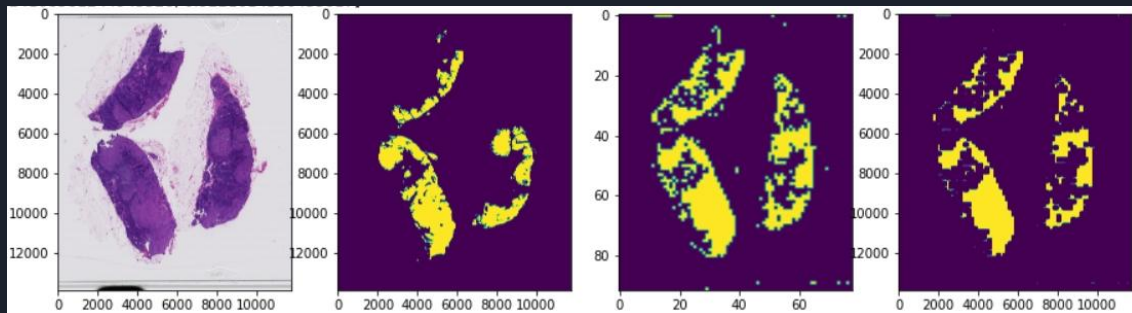
# Best Predictions from previous setting

Version c's prediction on img 64:



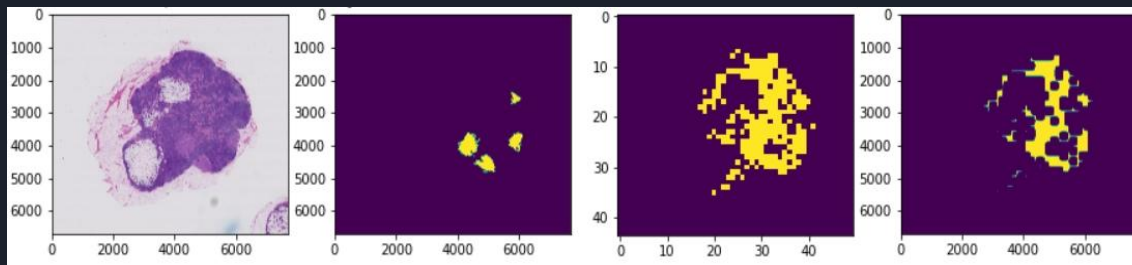
Version 2.a's


prediction on img 78:



Version 2.a's

Prediction on img 91:





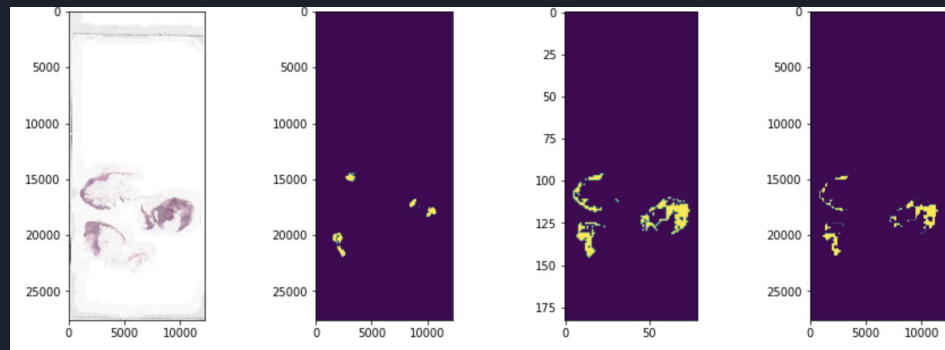
## 5.c Predictions (Using zoom level 2,3,4 & stride 299)

- There does not exist a single model that outperforms other models in predictions
- Could not use data augmentation from this example onwards due to RAM Problems.
- The reason why the true positive rate is so high for image 91 is because the model predicted a lot of Normal Patches to be tumors.
- The model is not so good in an accuracy point of view.

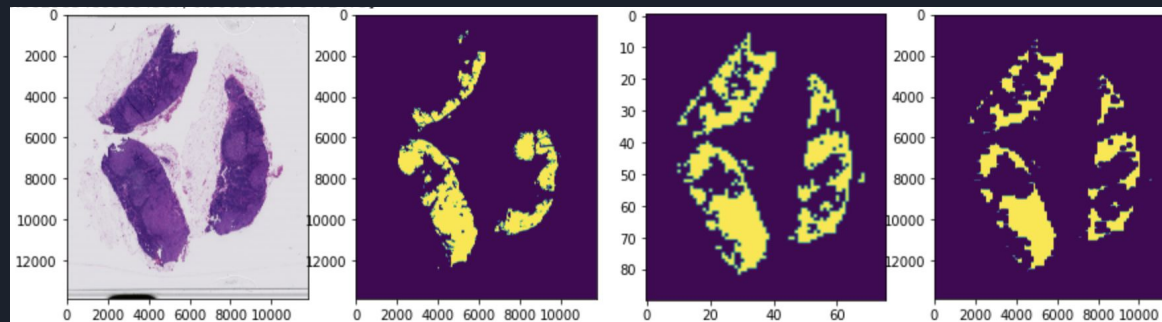
Image number	64	78	91
Best model	Version 2.b	Version 2.a	Version 2.a
False Negative Rate	0.31	0.30	0.06
True Positive Rate	0.69	0.70	0.94
AUC	0.84	0.88	0.88

# Best Predictions from previous setting

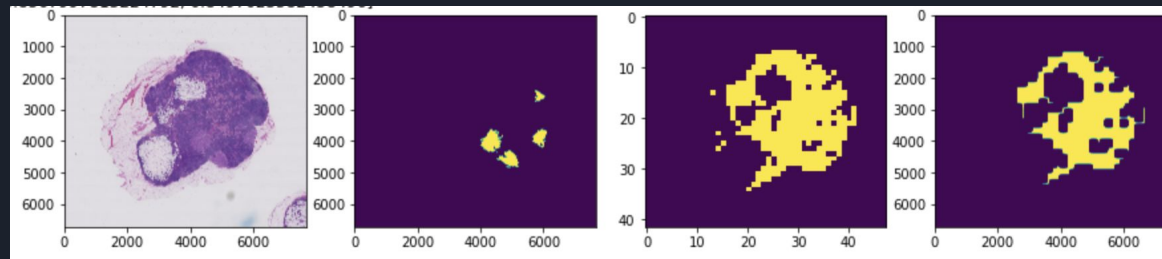
Version 2.b's prediction on img 64:



Version 2.a's  
prediction on img 78:



Version 2.a's  
Prediction on img 91:





## 5.d Predictions (Using zoom level 2,3,4 & stride 200)

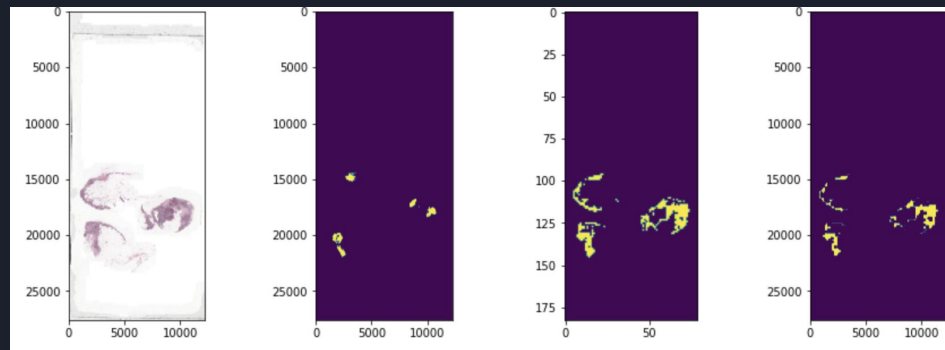
- There does not exist a single model that outperforms other models in predictions
- The reason why the true positive rate is so high for image 91 is because the model predicted a lot of Normal Patches to be tumors.
- The model is not so good in an accuracy point of view.
- The sample problem from the above example happened.

Image number	64	78	91
Best model	Version b	Version 2.a	Version 2.a
False Negative Rate	0.29	0.30	0.04
True Positive Rate	0.71	0.70	0.96
AUC	0.85	0.82	0.89

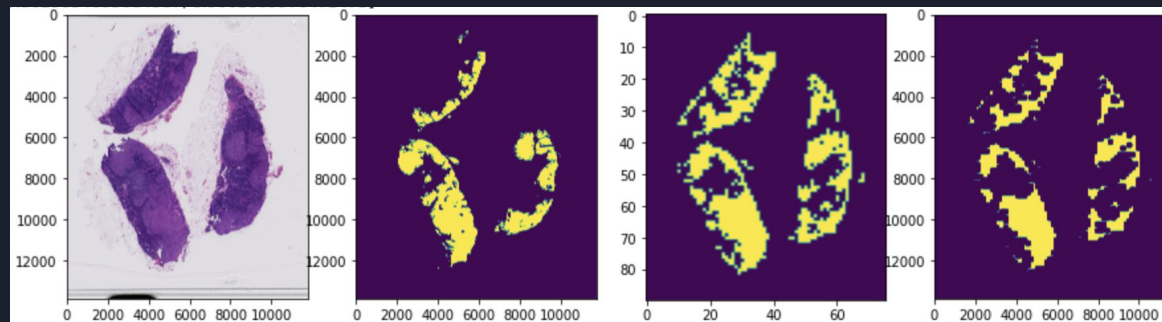


# Best Predictions from previous setting

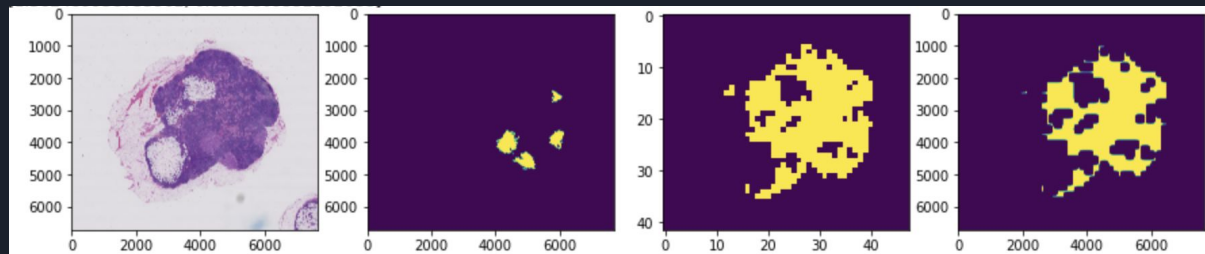
Version b's prediction on img 64:




Version 2.a's  
prediction on img 78:



Version 2.a's  
Prediction on img 91:





## 5.e Predictions (Using zoom level 1,2 & stride 299)

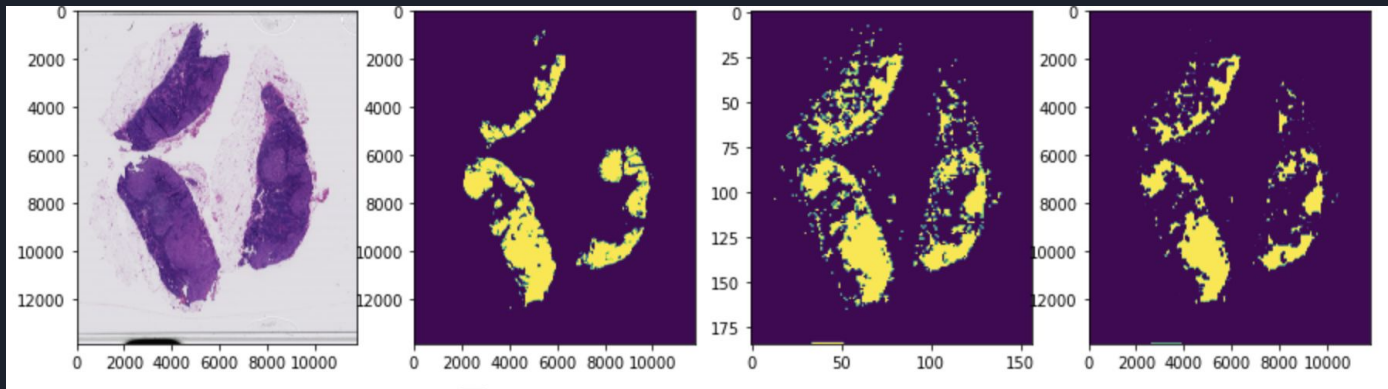
- Version b works well for both predictions.
- Although the scores are not too different from the models we used above, the model doesn't excessively predict patches to be tumorous.
- The predictions are the best we can see so far.
- Lowering zoom levels greatly helps the quality of the predictions.

Image number	64	78	91
Best model	RAM Issues	Version b	Version b
False Negative Rate	X	0.29	0.11
True Positive Rate	X	0.71	0.89
AUC	X	0.85	0.81

# Best Predictions from previous setting

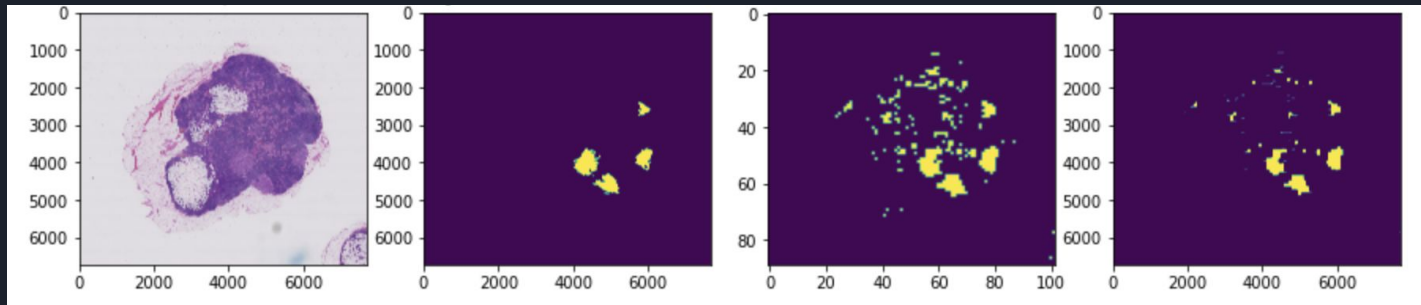
Version b's

prediction on img 78:



Version b's

Prediction on img 91:





## 5.f Predictions (Using zoom level 0,1 & stride 299)

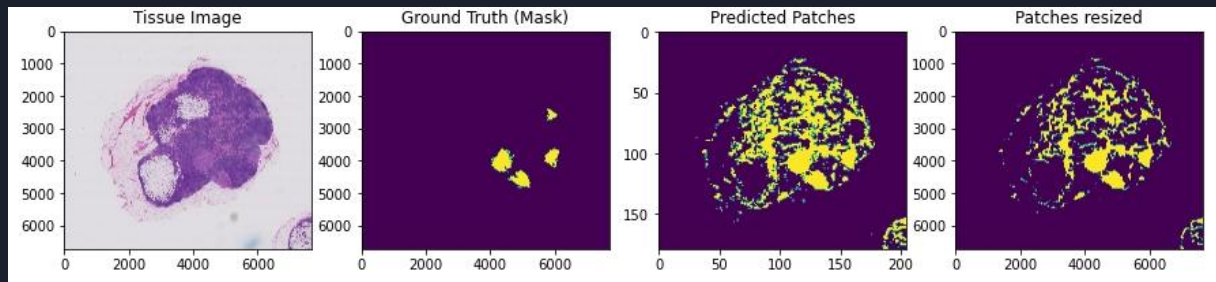
- For this specific case only, used image number 84,94,101,110 for the training set, 75 for validation set, and 91 for the test set (to work with limited Colab Pro's RAM size)
- Model that achieves highest accuracy does not produce most accurate heatmap

Image number	64	78	91
Best model	X	X	Version b
False Negative Rate	X	X	0.01
True Positive Rate	X	X	0.99
AUC	X	X	0.93

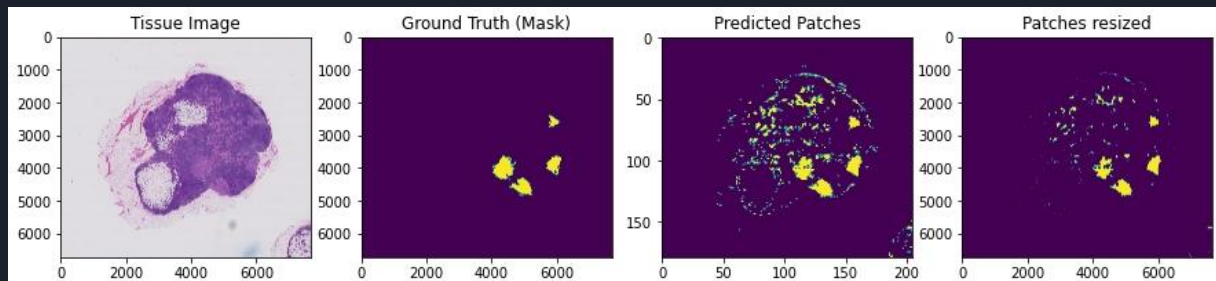
Image number	91
<b>2nd</b> best model	Version 2.a
False Negative Rate	0.22
True Positive Rate	0.78
AUC	0.87

# Best Predictions from previous setting

Best model: Version b's  
prediction on img 91:



2nd Best model: Version 2.a's  
prediction on img 91:





## 6. Conclusion

- In conclusion, we believe that the results depend mostly on the resolution of the images and what images we select for the train, test, valid sets.
- We could observe that the predictions vary on whether the image has a large amount of tumor patches or a small number of tumor patches.
- The zoom level was also a big factor in the quality of the predictions where a lower zoom level will produce a much better image.
- For detailed documentation and code explanation please check the github link below.
- [https://github.com/bluecube246/Tumor\\_Semantic\\_Image\\_Segmentation](https://github.com/bluecube246/Tumor_Semantic_Image_Segmentation)