Detecting Cancer Metastases on Gigapixel Pathology Images

Applied Deep Learning - Fall ‘19

Ankita Agrawal (aa4229), Sarang Gupta (sg3637)

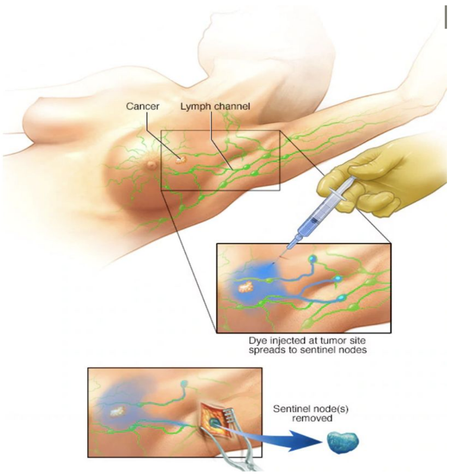
Table Of Contents

Detecting Cancer Metastases

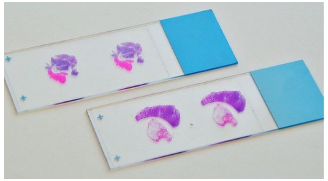
Problem Statement And Motivation

Detecting Cancer Metastases

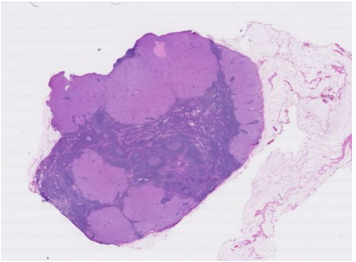
- Breast cancer is a **most common** and **deadliest** cancer spread across the world
- A key challenge for **pathologists** in assessing lymph node status is the large area of tissue that has to be examined to identify metastases which is both **time intensive** and **sensitive** process
- Sometimes the pathologists might even **miss small metastases**
- The goal of our solution is to create a **automated detection tool** to detect metastases in **whole-slide images** of lymph node sections from female breasts using deep learning

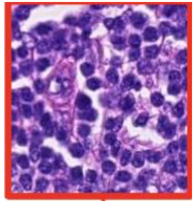


Biopsy

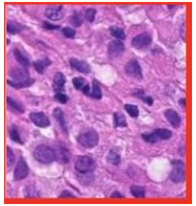


Inspection





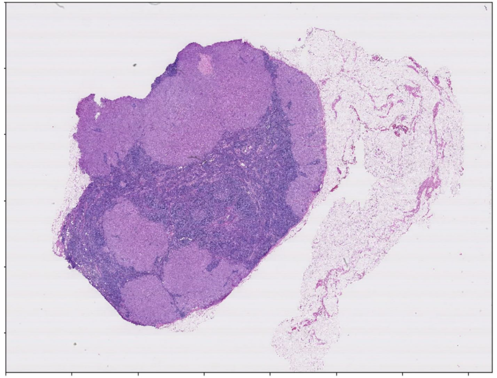
Healthy Cells



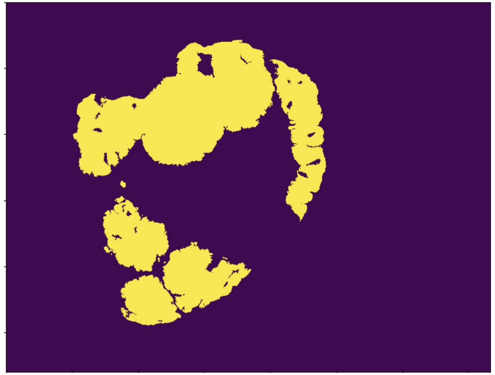
Tumor Cells

Description of Data

Data Source



Tissue Image



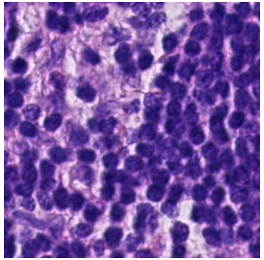
Mask Image

- Dataset comprises of whole slide images of **sentinel lymph nodes** of breast cancer patients
- Original dataset comprised of **400 images**. For computational simplicity we have subsampled **22 images**
- Each of these **22 images** have a mask which points to cancer cells located in the slide
- Each slide image is **~2GB** and mask is **~300MB**
- Each slide image can also be magnified up to **40x**
- As per the different zoom level each slide can be categorized into **8 levels** with 0 being the highest resolution (40x) and 7 being the lowest resolution

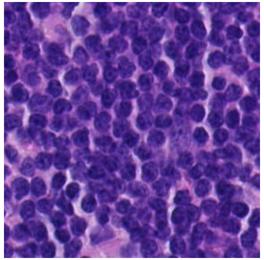
Description of Data

Dataset At Different Zoom Level

Level 0

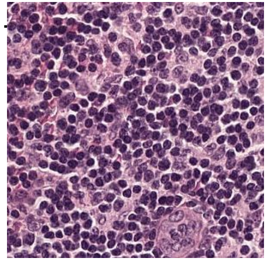


Healthy Cell

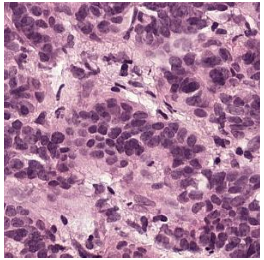


Tumor Cell

Level 1

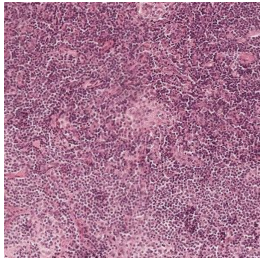


Healthy Cell

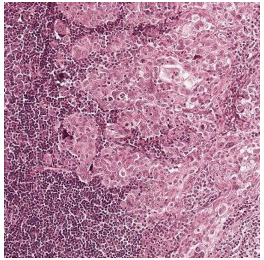


Tumor Cell

Level 3

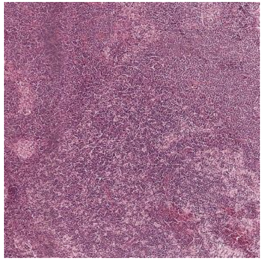


Healthy Cell

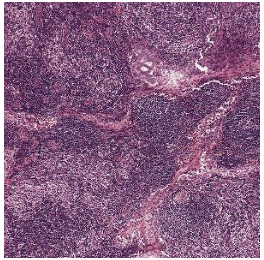


Tumor Cell

Level 4



Healthy Cell



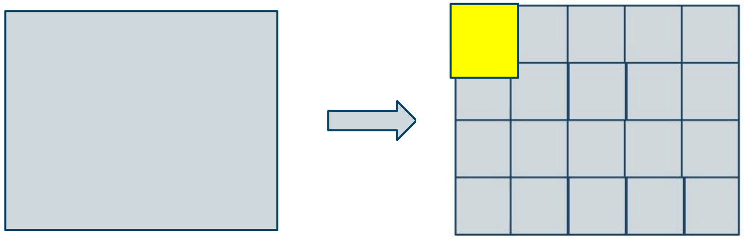
Tumor Cell

Methodology

Data Generation, Train Test Split, Metrics

Data Generation: Every image is processed using **open slide** library. Each image is further divided into smaller patches of image size **(299 x 299)** using a stride of 299 at zoom levels (level 0 - 5)

For ex:



Whole Slide Image **Small Patches**

- For levels 0, 1, 2 the image resolution is very high resulting in exponential no of patches, hence these patches are **randomly sub-sampled**
- Data sanity is maintained by selecting patches with at least **30% tissue cells**

Data Ingestion: Image Data Generator by Tensor FLow is used which generates batch of images (**batch size = 32** here) with real time data augmentation

The generator directly reads the image files from the file directory

Train Test Split

- Model is trained on **18** images and tested on **3** images
- The validation data comprises of random **20%** patches that are generated using **18** training images

Metrics

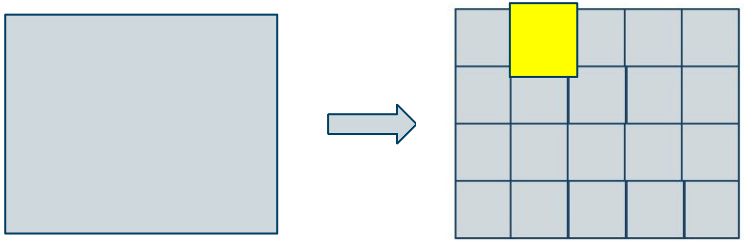
- Precision, Recall and AUC are used to judge model capability given that it is classification problem
- Heat Map depicting prediction probability is an another way which is used to test our model performance

Methodology

Data Generation, Train Test Split, Metrics

Data Generation: Every image is processed using **open slide** library. Each image is further divided into smaller patches of image size **(299 x 299)** using a stride of 299 at zoom levels (level 0 - 5)

For ex:



Whole Slide Image **Small Patches**

- For levels 0, 1, 2 the image resolution is very high resulting in exponential no of patches, hence these patches are **randomly sub-sampled**
- Data sanity is maintained by selecting patches with at least **30% tissue cells**

Data Ingestion: Image Data Generator by Tensor FLow is used which generates batch of images (**batch size = 32** here) with real time data augmentation

The generator directly reads the image files from the file directory

Train Test Split

- Model is trained on **18** images and tested on **3** images
- The validation data comprises of random **20%** patches that are generated using **18** training images

Metrics

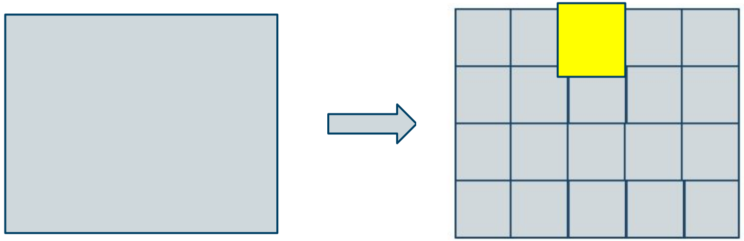
- Precision, Recall and AUC are used to judge model capability given that it is classification problem
- Heat Map depicting prediction probability is an another way which is used to test our model performance

Methodology

Data Generation, Train Test Split, Metrics

Data Generation: Every image is processed using **open slide** library. Each image is further divided into smaller patches of image size **(299 x 299)** using a stride of 299 at zoom levels (level 0 - 5)

For ex:



Whole Slide Image **Small Patches**

- For levels 0, 1, 2 the image resolution is very high resulting in exponential no of patches, hence these patches are **randomly sub-sampled**
- Data sanity is maintained by selecting patches with at least **30% tissue cells**

Data Ingestion: Image Data Generator by Tensor FLow is used which generates batch of images (**batch size = 32** here) with real time data augmentation

The generator directly reads the image files from the file directory

Train Test Split

- Model is trained on **18** images and tested on **3** images
- The validation data comprises of random **20%** patches that are generated using **18** training images

Metrics

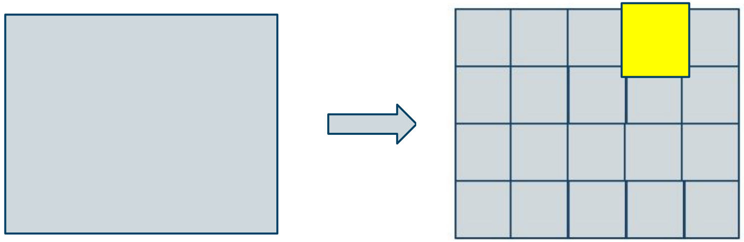
- Precision, Recall and AUC are used to judge model capability given that it is classification problem
- Heat Map depicting prediction probability is an another way which is used to test our model performance

Methodology

Data Generation, Train Test Split, Metrics

Data Generation: Every image is processed using **open slide** library. Each image is further divided into smaller patches of image size **(299 x 299)** using a stride of 299 at zoom levels (level 0 - 5)

For ex:



Whole Slide Image **Small Patches**

- For levels 0, 1, 2 the image resolution is very high resulting in exponential no of patches, hence these patches are **randomly sub-sampled**
- Data sanity is maintained by selecting patches with at least **30% tissue cells**

Data Ingestion: Image Data Generator by Tensor FLow is used which generates batch of images (**batch size = 32** here) with real time data augmentation

The generator directly reads the image files from the file directory

Train Test Split

- Model is trained on **18** images and tested on **3** images
- The validation data comprises of random **20%** patches that are generated using **18** training images

Metrics

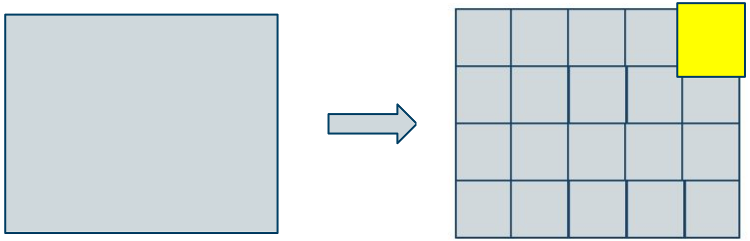
- Precision, Recall and AUC are used to judge model capability given that it is classification problem
- Heat Map depicting prediction probability is an another way which is used to test our model performance

Methodology

Data Generation, Train Test Split, Metrics

Data Generation: Every image is processed using **open slide** library. Each image is further divided into smaller patches of image size **(299 x 299)** using a stride of 299 at zoom levels (level 0 - 5)

For ex:



Whole Slide Image **Small Patches**

- For levels 0, 1, 2 the image resolution is very high resulting in exponential no of patches, hence these patches are **randomly sub-sampled**
- Data sanity is maintained by selecting patches with at least **30% tissue cells**

Data Ingestion: Image Data Generator by Tensor FLow is used which generates batch of images (**batch size = 32** here) with real time data augmentation

The generator directly reads the image files from the file directory

Train Test Split

- Model is trained on **18** images and tested on **3** images
- The validation data comprises of random **20%** patches that are generated using **18** training images

Metrics

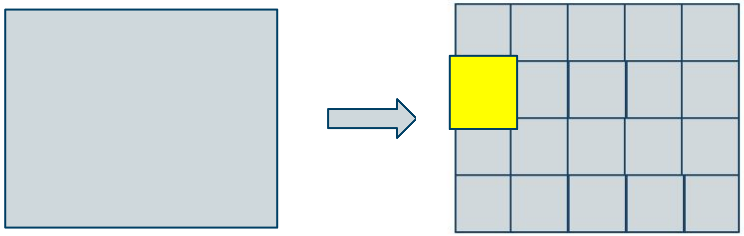
- Precision, Recall and AUC are used to judge model capability given that it is classification problem
- Heat Map depicting prediction probability is an another way which is used to test our model performance

Methodology

Data Generation, Train Test Split, Metrics

Data Generation: Every image is processed using **open slide** library. Each image is further divided into smaller patches of image size **(299 x 299)** using a stride of 299 at zoom levels (level 0 - 5)

For ex:



Whole Slide Image **Small Patches**

- For levels 0, 1, 2 the image resolution is very high resulting in exponential no of patches, hence these patches are **randomly sub-sampled**
- Data sanity is maintained by selecting patches with at least **30% tissue cells**

Data Ingestion: Image Data Generator by Tensor FLow is used which generates batch of images (**batch size = 32** here) with real time data augmentation

The generator directly reads the image files from the file directory

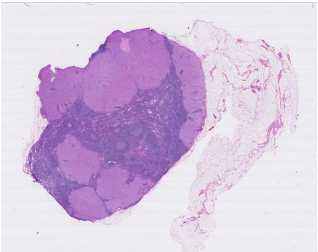
Train Test Split

- Model is trained on **18** images and tested on **3** images
- The validation data comprises of random **20%** patches that are generated using **18** training images

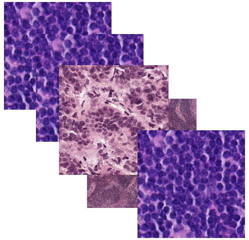
Metrics

- Precision, Recall and AUC are used to judge model capability given that it is classification problem
- Heat Map depicting prediction probability is an another way which is used to test our model performance

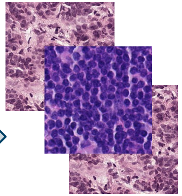
Methodology
Flowchart

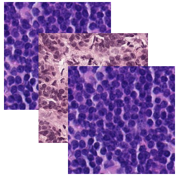


Whole Slide Image

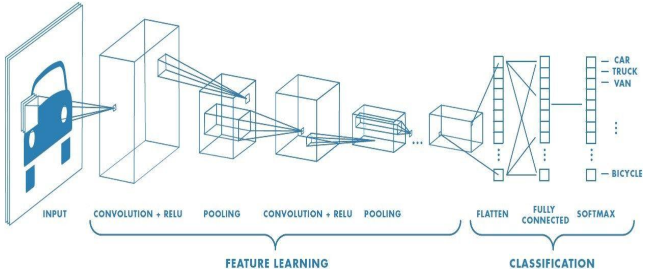


Patches

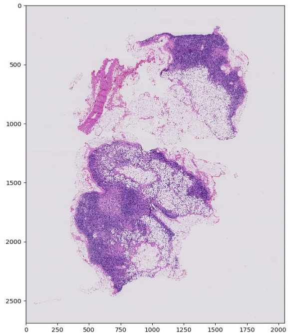




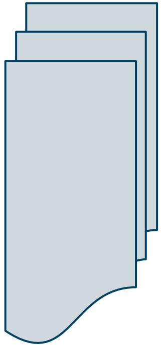
Training Batches



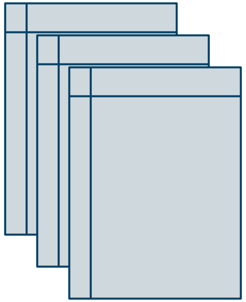
Inception Model as Base Model



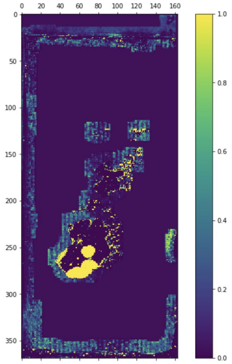
Test Image



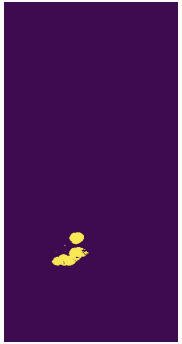
Deep Neural Network Model



Probability Matrix



Predicted Mask



True Mask

Modelling
Experimentation

Experiment at different zoom level

Zoom Level	Tumor Patches	Healthy Patches	AUC
1	6507	5925	0.9598
2	2006	5227	0.8624
3	16257	3143	0.8607
4	3982	1130	0.7413
3, 4, 5	974	460	0.7245

Experiment at a Model Level:

1. **Simple Architecture:** The baseline model is built using a simple neural network using 2 convolutional layers with max pooling and dropout. This model is underfitting so we moved onto a complex network

2. **VGG16 :** We experiment with pretrained model VGG16. This model gives a lesser recall and hence is not the best model to train on this data

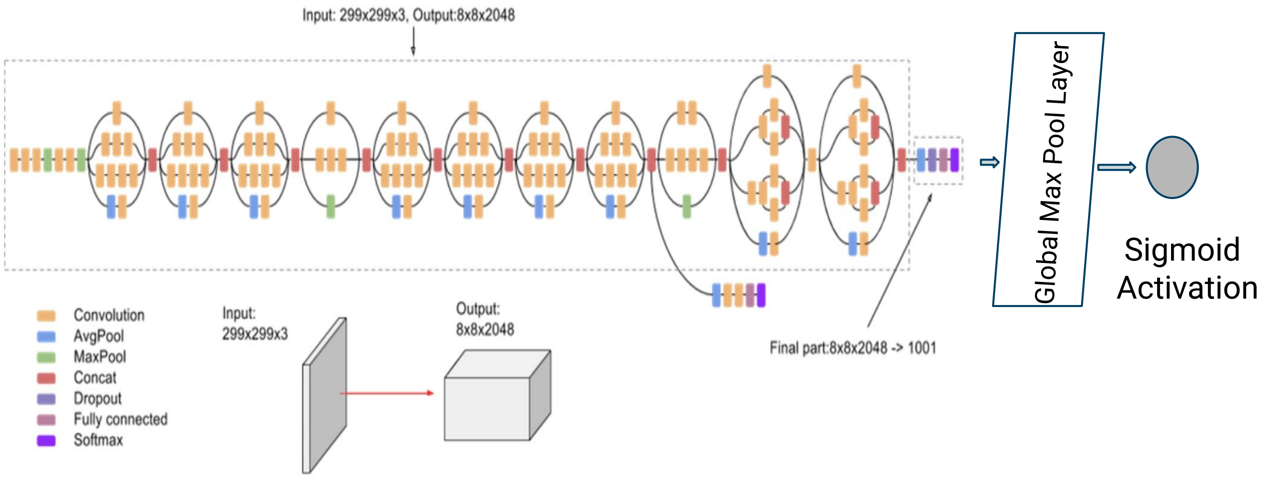
3. **InceptionV3**: The best performing model is pretrained model on 'imagenets' weight. This model outperforms other models in terms of precision, recall and auc

   We experiment with this model in following ways **Only on Level 1, Only on Level 2, and On Level 3, 4 and 5**. The results of which can be seen on the table here

Modelling

InceptionV3 As Base Model

InceptionV3 model is made up of symmetric and asymmetric building blocks, including convolutions, average pooling, max pooling, concats, dropouts, and fully connected layers. Batchnorm is used extensively throughout the model and applied to activation inputs.



- Pre trained Inception Model with **'imagenets'** weights are fine tuned by retraining
- The top layer of InceptionV3 model is replaced by **GMP** layer and **Dense** layer with **sigmoid** activation on 1 node (binary classification problem)
- Learning Rate of **0.001** is used with **RMSprop** optimizer

Modelling

Model Configuration

Hyperparameters

Optimizer	RMSProp
Learning Rate	0.0001
Rho	0.95
Epochs	10
Batch Size	32

Summary

```
Model: "sequential"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
inception_v3 (Model)         (None, 8, 8, 2048)        21802784
_________________________________________________________________
global_average_pooling2d (Gl (None, 2048)              0
_________________________________________________________________
dense (Dense)                (None, 1)                 2049
=================================================================
Total params: 21,804,833
Trainable params: 21,770,401
Non-trainable params: 34,432
_________________________________________________________________
```

Callback

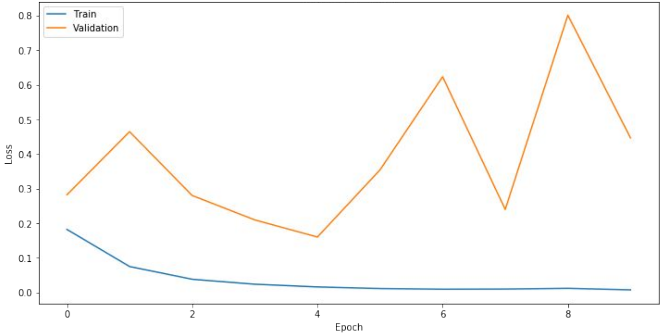
```
monitor='val_auc',
save_best_only=True, mode='auto',
save_weights_only=False
```

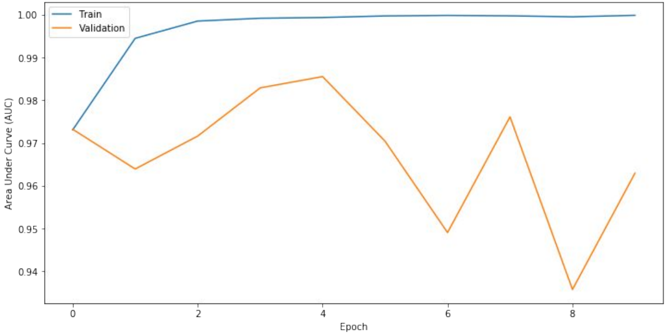
Class Weights

```
weight_for_0 = (1 / neg_count)*(neg_count+pos_count)/2.0
weight_for_1 = (1 / pos_count)*(neg_count+pos_count)/2.0
class_weight = {0: weight_for_0, 1: weight_for_1}
```

Modelling
Results

Level 1





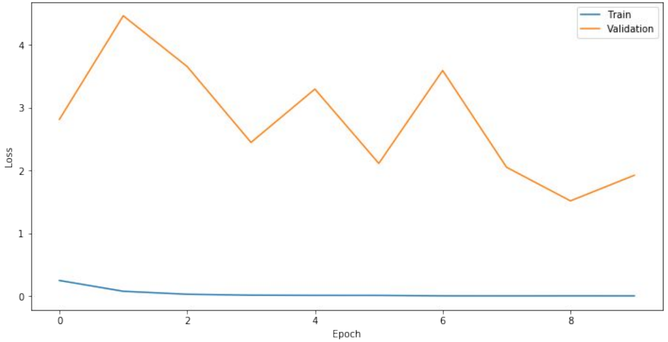
Best AUC: 0.9598

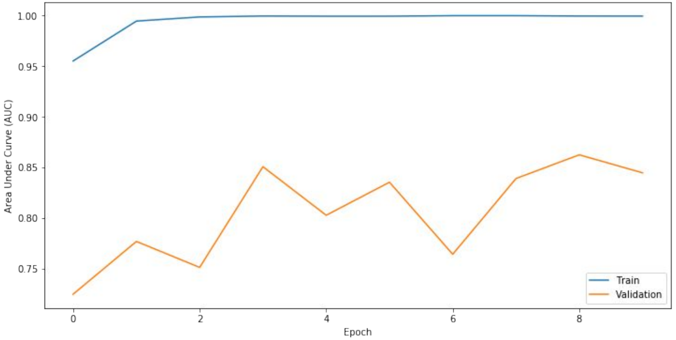
Precision: 0.9760

Recal: 0.9465

		Predicted	
		1	**0**
Actual	**1**	1221	69
	0	30	1144

Level 2





Best AUC: 0.8624

Precision: 0.8814

Recal: 0.5213

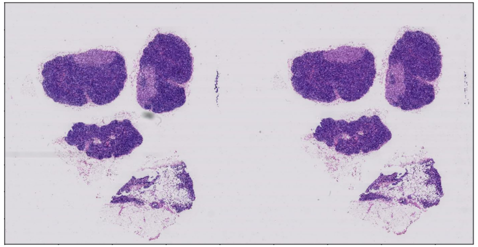
		Predicted	
		1	**0**
Actual	**1**	208	191
	0	28	1013

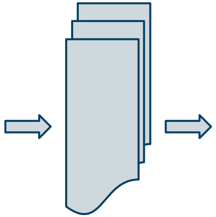
Generating Heatmap

Test Image 1

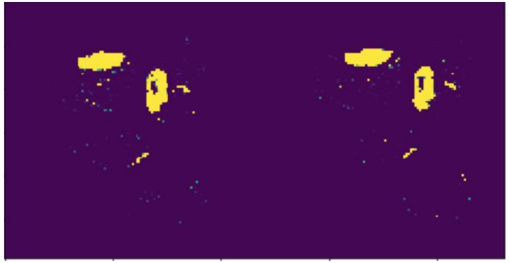
Predictions are done on individual 299 x 299 patches fed into the trained model. Output of the final dense layer are combined and plotted to generate heatmap.

Input

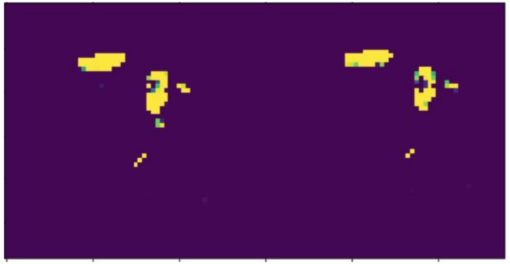




Output



Level 1



Level 2

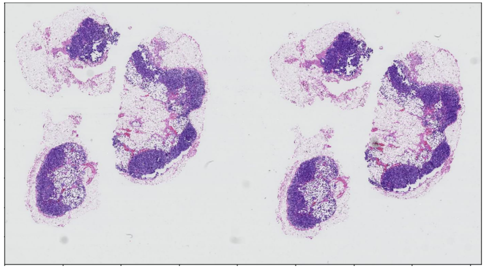
Actual

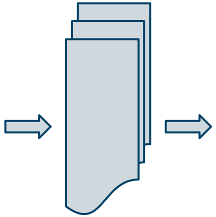
Generating Heatmap

Test Image 2

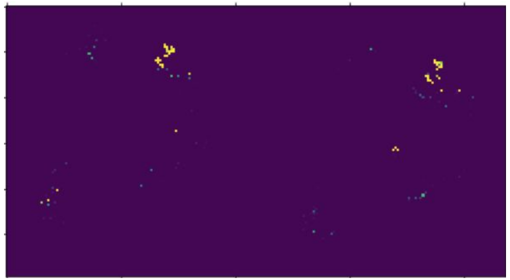
Predictions are done on individual 299 x 299 patches fed into the trained model. Output of the final dense layer are combined and plotted to generate heatmap.

Input





Output



Level 1

Level 2

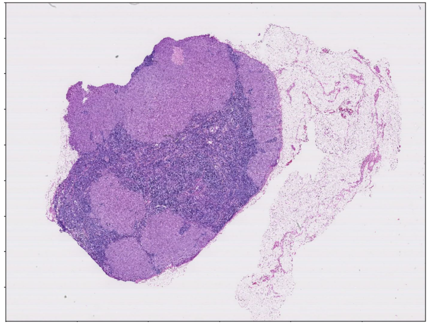
Actual

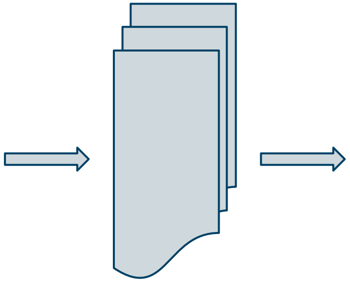
Generating Heatmap

Test Image 3

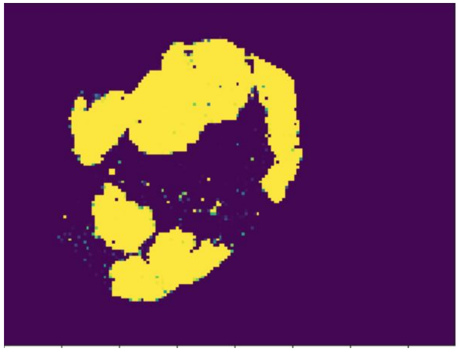
Predictions are done on individual 299 x 299 patches fed into the trained model. Output of the final dense layer are combined and plotted to generate heatmap.

Input

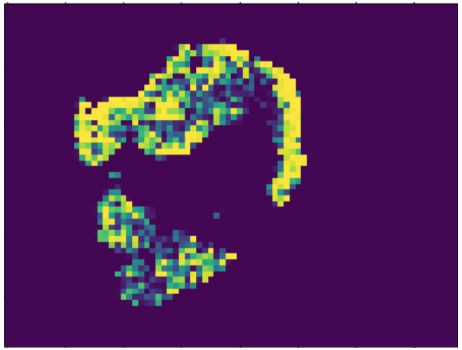




Output

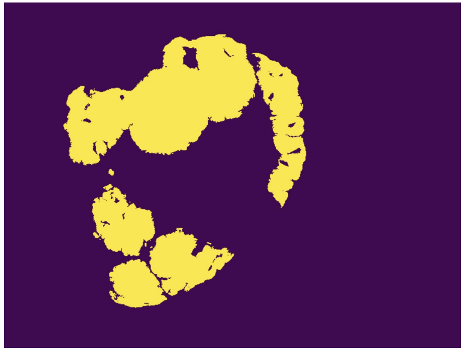


Level 1



Level 2

Actual



Model Usage And Discussion

An Assistive tool not a Replacement

- Our model can act as an **assistive tool** for pathologists
- The success of our model (high precision, recall and auc) holds a great promise to reduce the workload of the pathologists while at the same time **reduce the subjectivity in diagnosis**
- This model is of a **high clinical relevance** especially for organizations with **limited resource capabilities**
- It can be placed as a **first line of defence to help the smaller organizations** diagnose the underlying disease timely if it may exist

Limitations And Challenges

An Assistive tool not a Replacement

- Each image is in Gigabytes with upto **10^6 x 10^6 pixels** which are difficult to process at once
- Extracting patches on zoom level 0 for a single images takes about **~40-60 minutes** Training requires high computational power. It would take upto **3- 6 hours** to train a dataset of **15k** images on a single zoom level
- Due to lack of accessibility to private computing resources on cloud, our models are trained using publicly available **Google Colab**
- Colab only provides **~10 hours** of GPU access in a day which incurred lot of wait time
- Number of **read** and **write operation** on a Google Drive directory are also limited. This costed us additional efforts and time on creating duplicate directories for seamless model training
- It took several iterations of model **training** and **fine tuning** to figure out the **right threshold values** for the minimum tissue cell percentage. Patches below the threshold were omitted out of the dataset

Future Scope

Data for Good

- Scale this model to train with a **larger dataset** and predict on additional unseen images

- Build an **end-to-end pipeline** or **web tool** where doctors can input the patient's slide image and receive a prediction and mask image all in real time. This would help and guide the doctors towards a better inspection

- Create **awareness** about this tool and technology in the **cancer community**. Many might think it to be unreliable thus it is important to spread this tool as an automated assistance which comes at no cost and overhead

Acknowledgement

We would like to extend our gratitude to our course instructor **Prof. Josh Gordon** and the TAs for their support and guidance throughout the course of the semester and the project

References

- https://arxiv.org/abs/1703.02442
- https://camelyon16.grand-challenge.org/Home/
- https://openslide.org/
- https://github.com/openslide/openslide-python/tree/master/examples/deepzoom


THANK
YOU