Determining Cancer Hormone Receptor Status from H&E Stained Tissue Using Deep Learning

Dean Kelley

Committee Members
Dr. Anderson Nascimento Professor, Tacoma School of Engineering and Technology, Chair
Dr. Martine De Cock Professor, Tacoma School of Engineering and Technology
Dr. Juhua Hu Professor, Tacoma School of Engineering and Technology

UNIVERSITY of WASHINGTON

Background

- Every year
  - Millions of women are diagnosed with breast cancer
  - Hundreds of thousand die
- A large majority of invasive breast cancers are hormone receptor-positive (HR+)
  - Tumor cells grow in the presence of estrogen (ER) and/or progesterone (PR)
  - Hormone receptor status (HRS) is a key molecular marker used for prognosis and treatment decisions
  - HRS is determined by pathologists by: Immunohistochemistry (IHC) stained biopsied tissue for the targeted receptor
  - This highlights the presence of cellular surface antigens

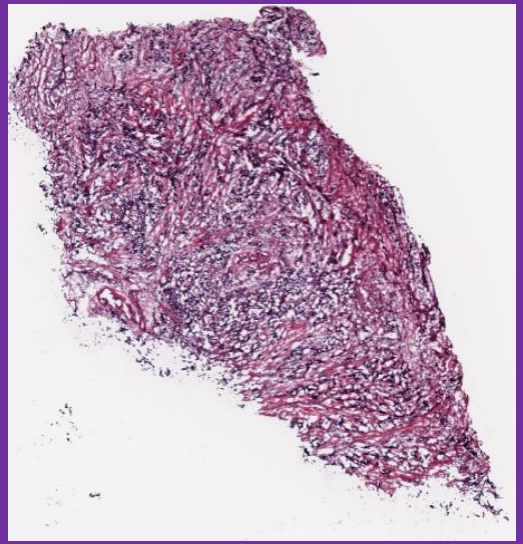
So, what's the problem?

Traditional HRS determination is:

- Expensive
- Time consuming
- Inconsistent due to IHC preparation and pathologist subjectivity

Is there an alternative?

- Hematoxylin and eosin (H&E) staining
  - Highlights cellular morphology
  - Quick
  - Less expensive
  - Less variable in preparation.
- This gives us H&E-stained whole slide images (WSIs)

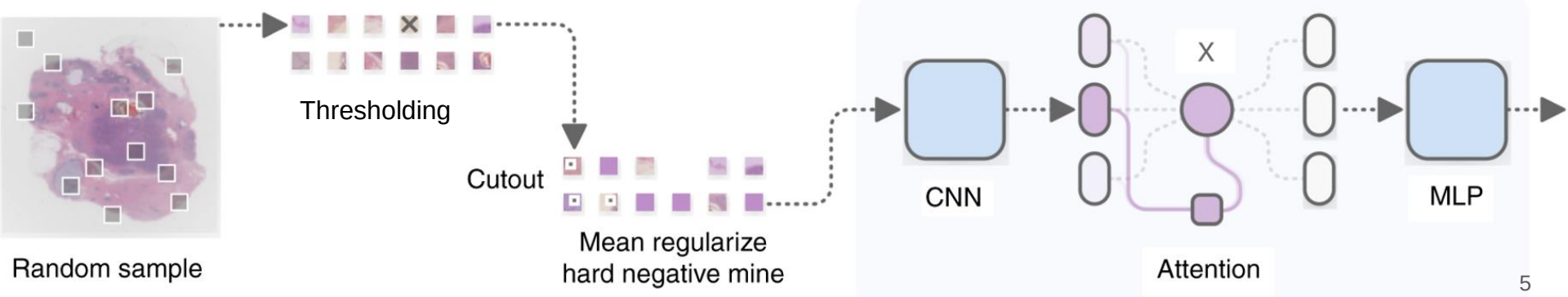


Solution

- Use machine learning to determine receptor status (RS) from H&E WSIs
- Morphology of the tumor, captured in the H&E stain, contains predictive signal for the molecular marker status of the tumor
- A machine-learning (ML) model can directly determine RS
- Model is trained with clinical RS that are available from patient records and requires no additional pixel-level annotations
- Train models using H&E WSIs as input data, and IHC annotations as input data labels
- Use model for future WSI RS inference

How

- Design an attention-based deep neural network that uses multiple instance learning (MIL), which we are calling ReceptorNeXt
- ReceptorNeXt is trained to predict RS from a bag of tiles randomly selected from a WSI
- The WSI is divided into 256 × 256 pixel-size nonoverlapping tiles.



Source: Naik et al

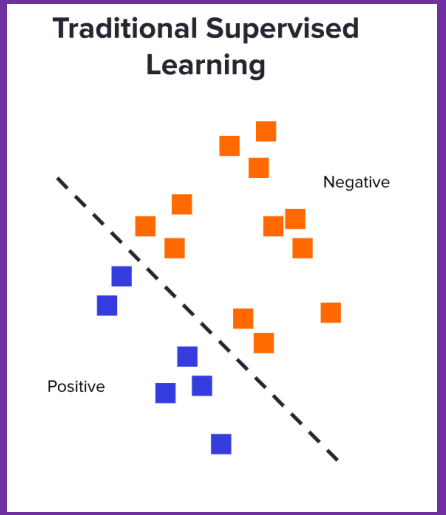
What is Multiple Instance Learning (MIL)?

- Bag-level supervised machine learning that uses sets of labeled bags to learn a classifier
- A bag contains a set of data, such as images, and is assigned a binary label
- Label is based on the presence or absence of data with a particular property.
  - A bag is labeled 'positive' if it contains at least one positive instance
  - ‘Negative' if otherwise

MIL Continued - MIL Compared to Traditional ML

Traditional ML

- Each example is represented by a single feature vector and has a corresponding label
- Features are extracted from the individual instances.
- Try to find the details that separate positive and negative cases

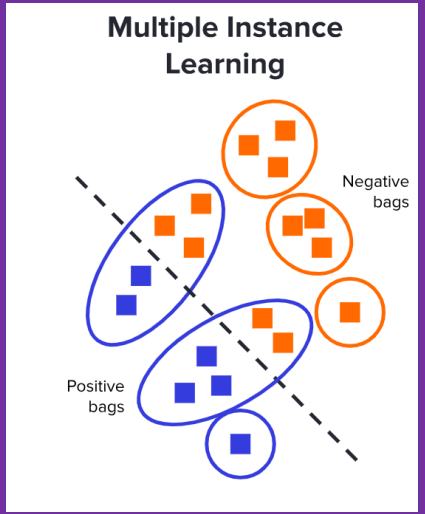


https://nilg.ai/202105/an-introduction-to-multiple-instance-learning/

MIL Continued - MIL Compared to Traditional ML

MIL ML

- We may not know whether a specific instance is positive or negative
- We do know that at least one instance must be if a bag is labeled positive
- Examining the instances in each bag can allow us to find the patterns that signify a positive instance



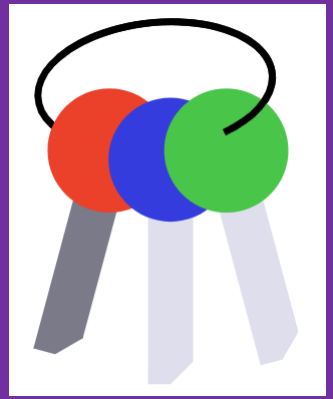
https://nilg.ai/202105/an-introduction-to-multiple-instance-learning/

MIL Continued - MIL Example

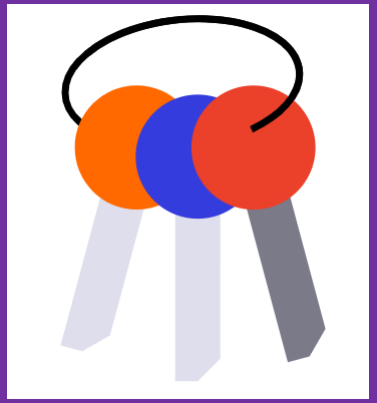
Which key opens the box?

You will be the model and I will feed you the data

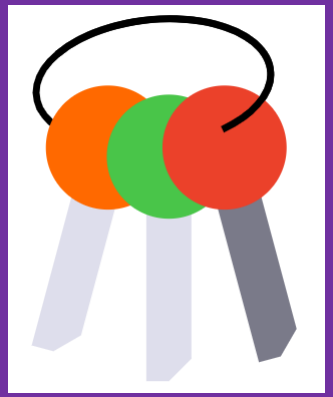
MIL Continued - MIL Example



Positive

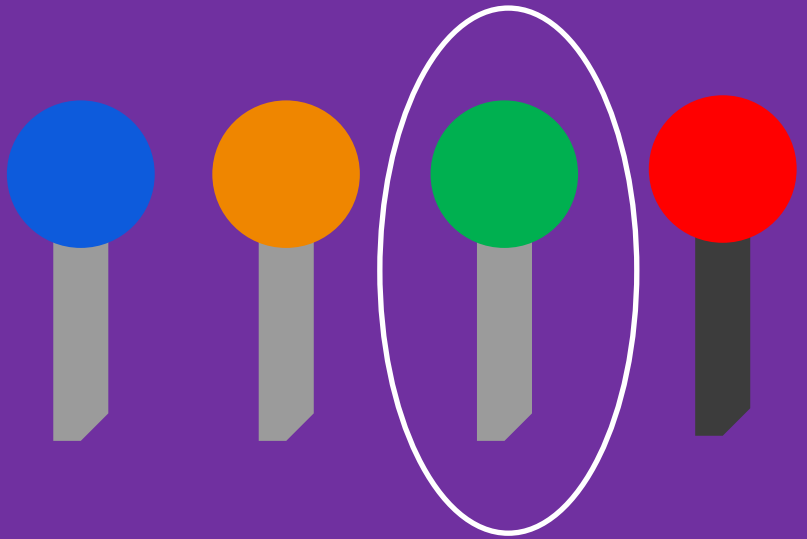


Negative



Positive

MIL Continued - MIL Example



Data

3111 images from 1098 patients from The Cancer Genome Atlas (TCGA) dataset
Each WSI has three receptor classifiers:

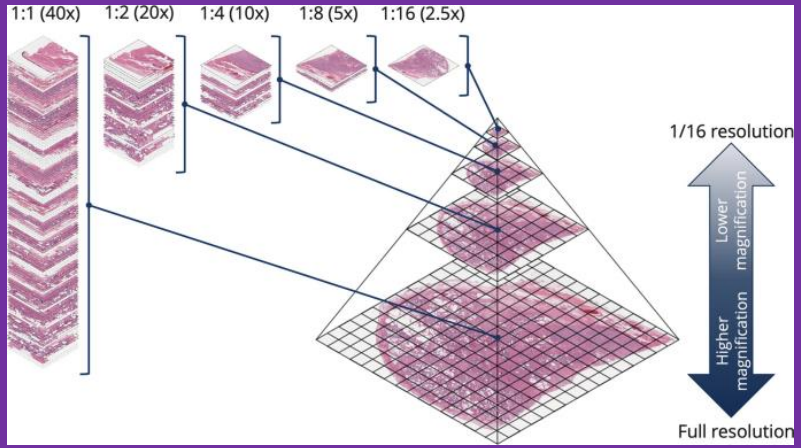
1. Estrogen (ER)
2. Progesterone (PR)
3. Human Epidermal Growth Factor Receptor 2 (HER2)

Each of these have the following possible statuses:

- Positive
- Negative
- Equivocal (ambiguous)
- Null

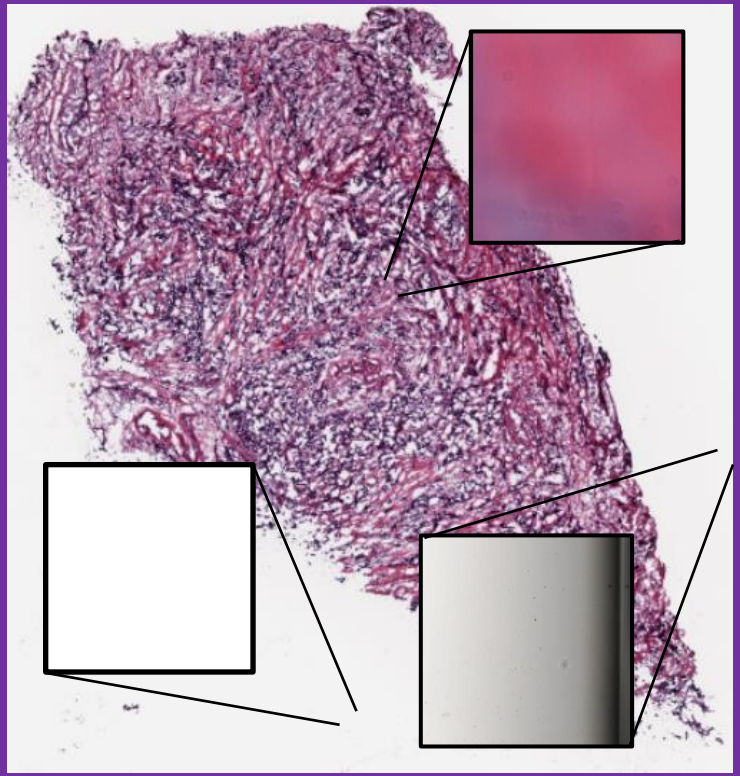
Data

- Images range in dimensions of 30k-200k pixels high and/or wide and range in size 300MB-3GB
- Some images are captured in 20x and 40x zoom
- Images are saved in the form of multiple levels of resolution



Data Processing

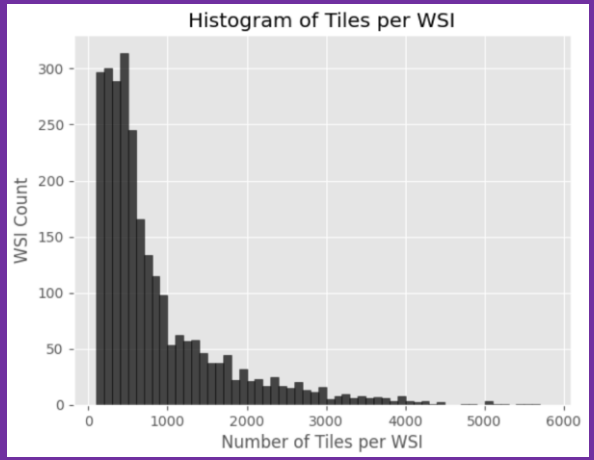
- Used OpenSlide package in Python to acquire images at 20x zoom and break into 256x256 pixel tiles
- Utilized multiprocessing to parallelize the tile saving process
- Removed tiles containing:
  - Predominantly white space
  - Out of focus/blurry tissue
  - Annotations or empty data that gets past the above
- Removed WSI containing fewer than 100 tiles



Processed Data

- For each respective HR, WSIs containing equivocal and null entries are removed
- A fourth classification was produced from the data – Triple Negative (NNN)
  - NNN is positive if all ER, PR and HER2 are negative
  - Else, negative

HR	Count	Percentage	
		Positive	Negative
ER	2516	76.6	23.4
PR	2509	65.8	34.2
HER2	1762	22.5	77.5
NNN	1747	16.1	83.9



The Model

ReceptorNeXt, consists of three interconnected neural networks which are trained together

- A feature extractor which converts each 256 × 256 pixel-size tile in a bag into a 512-dimensional feature vector
- An attention module that creates a 512-dimensional aggregate of feature vectors from all the tiles in the bag, attention-weighted based on discriminative power, and
- A decision layer that computes the probability of this bag being positive from the aggregate feature vector.

Method

- Perform 10-fold cross-validation on ERS
- Run experiments with:
  - Various feature extractors
    - ResNet50, ResNet101
    - ResNext50, ResNeXt101
  - Implementing Squeeze and Excitation blocks
  - Automatic Mixed Precision (AMP)
  - Ensemble Methods
- Utilizing best model configuration, run inferences
  - PRS
  - HER2S
  - NNNS

Method

- Cross-Validation of Model:
- 300 epoch per fold
- Early stopping
- AUROC as performance metric
  - Data is very unbalanced, but using to compare with reference paper
- Test model on test dataset
- Ensemble Evaluation:
- Run n evaluations utilizing bags of x tiles
- Save inference results to dataframe
- Select the max value of each WSI inference and calculate AUROC
- Test 5 times and report average
- Compute difference from the baseline

Results

Automatic Mixed Precision

Precision	Runtime (min)							Max Memory Allocated (MiB)
	Train	Evaluation						
	50	50	100	200	300	400	500	
FP32	6:26	0:27	0:55	2:26	4:10	5:44	10:10	10477
AMP	3:04	0:15	0:30	1:28	2:26	2:38	3:22	9114

2x faster training
Over 3x faster evaluation with 500 tile bag size
Almost 1GB less memory used

Results

Feature Extractors

Model	Fold AUROC			Run Time (min)*	
	Avg	Min	Max	Train	Validate
ResNet50	0.774	0.643	0.853	2:17	1:27
ResNeXt50	0.798	0.76	0.839	2:35	1:34
ResNeXt50 w/ SE	0.818	0.786	0.866	3:04	1:46
ResNet101	0.806	0.775	0.85	3:15	2:03
ResNeXt101 w/ SE	0.746	0.716	0.782	4:27	2:28

* Per epoch

- ResNext50 w/ SE performed best
- Surprisingly, ResNeXt101 w/ SE performed poorly despite promise from ResNet101

Results

HR Inference *

Hormone Receptor	Cross-Val Avg. AUC	Test AUC
ER Baseline	0.86	N/A
ER	0.789**	0.781
PR	0.734	0.76
HER2	0.65	0.612
NNN	0.731	0.582

* 5-Fold Cross-Validation
** ER performed worse than baseline (Naik et al)

Results - Ensembles

ER

Bag Sizes	AUC Improv. MAX	Time
50 x 10	-1.65	5:00
100 x 5	-1.05	4:46
250 x 2	-0.2	4:10
500 x 1	-	3:31

PR

Bag Sizes	AUC Improv. MAX	Time
50 x 10	-2.40	5:08
100 x 5	0.00	4:46
250 x 2	1.00	4:13
500 x 1	-	3:31

Results - Ensembles

HER2

Bag Sizes	AUC Improv. MAX	Time
50 x 10	3.20	3:59
100 x 5	2.40	3:28
250 x 2	1.30	3:06
500 x 1	-	2:36

NNN

Bag Sizes	AUC Improv. MAX	Time
50 x 10	5.10	5:46
100 x 5	4.70	5:04
250 x 2	4.40	4:44
500 x 1	-	3:58

Conclusions

- Produced open source
  - Data processing pipeline
  - MIL model that can make inferences on HRSs
- AMP makes the model significantly more efficient
- The ensemble method improves AUC by several points for datasets with imbalanced dataset where positive cases are fewer

Items to Explore in the Future

- Acquire more data from
  - The Australian Breast Cancer Tissue Bank (ABCTB)
  - NIC Brazil which is building an expanded set of data with classifications for rare cancer subtypes
- Explore Nvidia DALI to improve data load times
- Explore Dual-Path networks utilizing ResNeXt and DenseNet feature extractors to extract useful information
- Utilize a multiple GPU cluster to parallelize the training process
- Configure model to run in Docker container for quick and easy deployment

Access

Project Repository:
https://github.com/deanak1987/TCSS702_Capstone
Data:
https://portal.gdc.cancer.gov/repository
Report:
https://github.com/deanak1987/TCSS702_Capstone/blob/main/DeanTCSS702CapstoneProjectReport2024.pdf

References

Project Inspiration
https://www.nature.com/articles/s41467-020-19334-3#Ack1