

Metastatic Tissue Classification with Machine Learning and Deep Learning Models

Debapriya Tula

Teresa Liang

Udbhav Avadhani




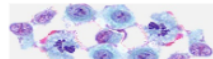


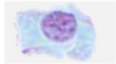

Abstract

Histopathologists typically analyze microscopic images of lymph nodes to determine the presence of cancerous cells, but this process is subjective. This project explores machine learning and deep learning techniques for objective metastatic tissue classification using PatchCamelyon, a publicly available dataset of histopathologic scans.

1 Introduction

The chances of curing someone from cancer are primarily in its detection and diagnosis, so an important area to focus on is the lymph nodes [4]. Our lymph nodes can swell and grow larger, which is a sign that the body is fighting an infection or illness. By examining microscopic images of lymph nodes, doctors can gain crucial insights into which patients have cancer.

Table 1: Differences between Normal and Cancerous Cells

Normal Cells	Cancerous Cells	Description of Cancerous Cells
		large and variably shaped nuclei
		many dividing cells and disorganized arrangements
		variation in size and shape of nuclei
		loss of normal feature (shape and morphology)

Histopathologists normally look at the shape, size, and distribution of cells and cell structures to determine if cells are cancerous. However, because the manual identification of cancer from histopathologic scans is subjective in nature and may vary from expert to expert, cancer remains a difficult task even with these advanced imaging techniques. This has caused many to look to AI to help them identify cancerous cells.

2 Objective

Using a publicly available dataset of histopathologic scans of lymph node sections, we sought to explore and evaluate various machine learning and deep learning techniques for classifying metastatic tissue.

3 Dataset

The dataset PatchCamelyon [11] included 327,680 color images (96×96 px) extracted from histopathologic scans of lymph node sections, with a rough 50-50 split of normal tissue and metastatic tissue. There were 4 blank images (all-white) that we filtered out before the analyses were done.

The specific dataset can be found at:
<https://www.kaggle.com/datasets/andrewmvd/metastatic-tissue-classification-patchcamelyon>.

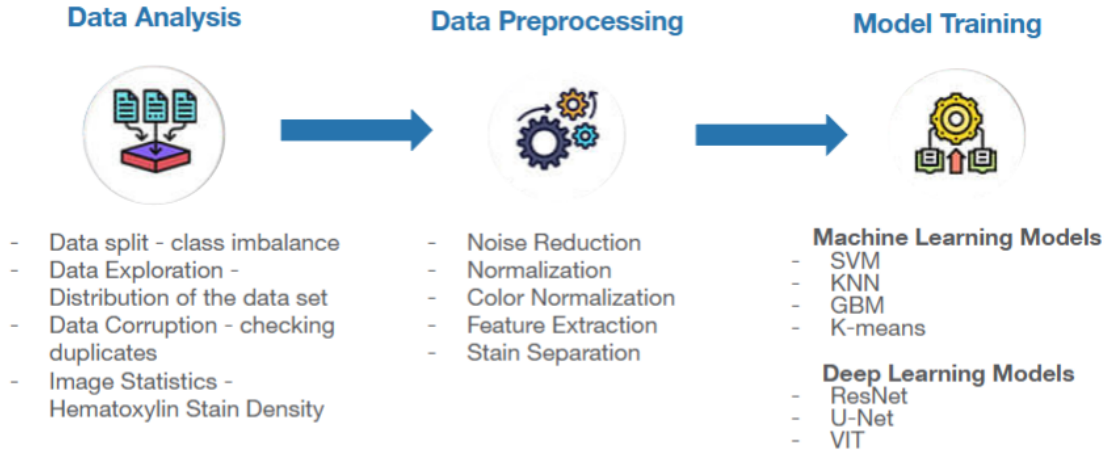


Figure 1: Procedural workflow for this project

4 Data Analysis

For data analysis, we examined the distributional characteristics of the positive and negative samples. Initially, our goal was to determine whether there was a significant difference between these two distributions. To achieve this, we employed the Euler characteristic function [3], which maps images to numerical values that can be used for assigning probability values.

The Euler characteristic is a topological measure that helps describe the shape or structure of an object. In simple terms, it provides a single number that characterizes an object’s connectivity, holes, and enclosed regions. For our case, we first binarize the image, and subtract the number of holes to the number of connected components to estimate an Euler characteristic for the histopathological images.

Now, when we have assigned a single value to each of the positive and negative images, we then apply the Kolmogorov Smirnov Test to see if the two distributions are the same. We observe that the p-value for this test turns out to be 1.49×10^{-6} , which shows that they are significantly different.

We also observe the distribution of intensities for positives and negatives 2a for 1000 samples and observe a significant difference in average intensities for the two classes. However, we do not capitalize on this as the intensities in the test set (and in real world scans) can be skewed which may lead to wrong predictions.

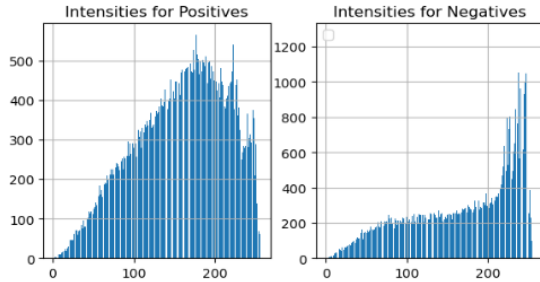
5 Data Preprocessing

We explored various preprocessing techniques such as normalization, color normalization, augmentation, and feature extraction. Key techniques included Contrast Limited Adaptive Histogram Equalization (CLAHE), Reinhard Normalization, and various image transformations. Many of these were adapted from scikit-image’s processing functions library [10]. For CLAHE, we used the OpenCV library [5]. In addition, we did not include several pre-processings in this paper, but they are documented in our final slideshow presentation.

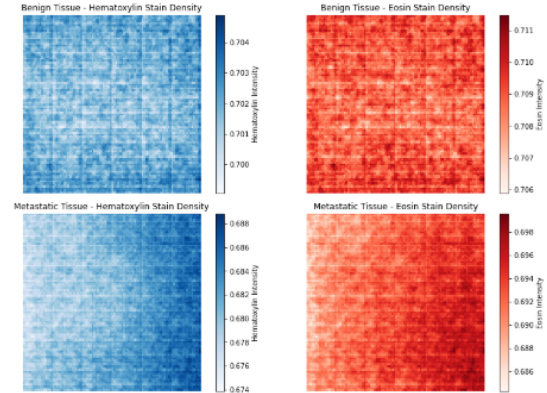
6 Machine Learning Models

Due to limited computational resources, we trained our machine learning models on only 1% of the training data, comprising 2,621 images. Since the training dataset has an exact 50-50 split between normal and metastatic tissue, we randomly selected indices from the original dataset using a fixed random seed to ensure reproducibility.

We experimented with three supervised machine learning models: a gradient boosting classifier (a boosting machine learning algorithm which tries to reduce the errors by weighing samples based on their gradients),



(a) Intensity distributions for positive (left) and negative (right) images



(b) Hematoxylin & Eosin Stain Density Distributions: In the benign tissues (above), we see uniform, well-organized cell patterns with both the Hematoxylin stain and the Eosin stain. In the malignant tissues (below), there seems to be a gradient where the intensity of hematoxylin and eosin density increases as you go from left to right.

Figure 2: Dataset distributions

a k-nearest neighbors classifier, and a support vector machine classifier. Additionally, we tested each preprocessing method listed in the table above individually with the models and explored various combinations of preprocessing techniques.

In addition, we also performed mini-batch K-means clustering on the data, which given that it is an unsupervised algorithm, we cannot compare to the supervised algorithms and instead compared silhouette and K-means scores between different numbers of clusters.

All of the machine learning algorithms were adapted from the scikit-learn library [7].

7 Deep Learning Models

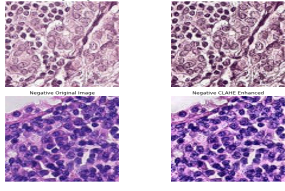
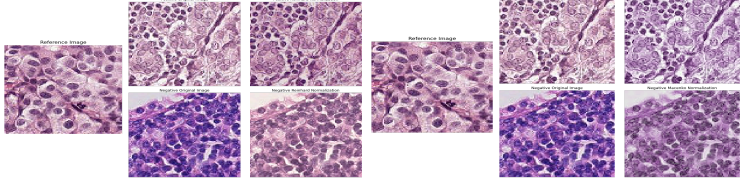
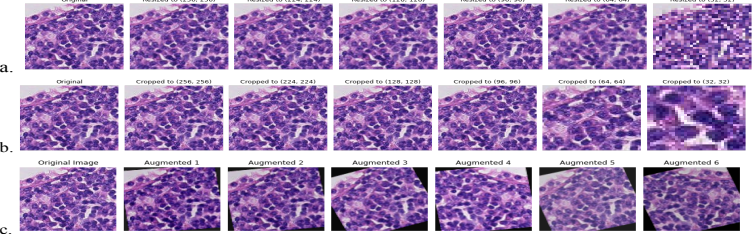
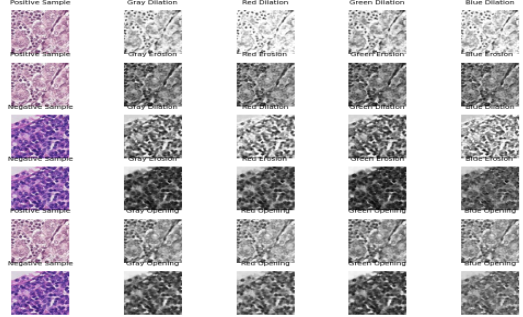
We primarily explored three vision architectures for the classification problem. For each of them, we train their base versions (with standard preprocessing) to start with. We also train them with each of the preprocessing techniques discussed above which have been found to be visually discernible.

The first architecture we explored was the ResNet-50 [2], a 25.6 M parameter model which was introduced by Microsoft in 2015. It was the first convolutional neural network which allowed having a deep network with a large number of layers by introducing residual connections preventing the vanishing of gradients. For our experiments, we choose the original Resnet50 with ImageNet pre-trained weights. For training the model, we use a batch size of 128, and train it for 20 epochs with an AdamW(learning_rate=1e-3, weight_decay=0.1) optimizer.

Using the same hyperparameter settings, we also explored a model called UNet [8], a 34.6 M parameter model popularly used for segmentation tasks. It consists of a contracting path which downsamples the image, and an expanding path which upsamples the downsampled features. Skip connections are introduced here as well to cater for vanishing gradients. We modify the UNet for our task of classification by adding a global average pooling (GAP) layer and classification head.

To understand how transformer based models would perform for this dataset, we trained a Vision Transformer (ViT-Base), an 86 M parameter model with our dataset. Vision transformer [1] follows the same attention mechanism as the language based transformer model. Tokens in case of images are however formed by cropping the image into patches (eg 16x16) and passing them through an embedding layer. We use the extra CLS token affixed to the tokens for classification. The model is trained for 10 epochs with a learning rate of 1e-4 and weight_decay of 0.01.

Table 2: Preprocessing techniques

Technique	Example
Normalization a. Contrast Limited Adaptive Histogram Equalization (CLAHE): improves contrast adaptively across different regions	 <p>The image shows four histology slides arranged in a 2x2 grid. The top row is labeled 'Positive Original image' and 'Positive CLAHE Enhanced image'. The bottom row is labeled 'Negative Original image' and 'Negative CLAHE Enhanced image'. The CLAHE-enhanced images show significantly improved contrast and detail compared to the original images.</p>
Color Normalization a. Reinhard Normalization: matches the color distribution of an image to a reference b. Macenko Method: utilizes stain deconvolution to normalize hematoxylin and eosin (H&E) staining	 <p>The image shows two sets of histology slides. Set 'a' illustrates Reinhard Normalization, where multiple images are shown alongside a 'Reference image' to show color matching. Set 'b' illustrates the Macenko Method, showing 'Original image' and 'Normalized image' for several samples, demonstrating consistent color across different slides.</p>
Augmentation a. Resizing: decreases resolution of image b. Cropping: Crops to a smaller part of the image c. Combination Transformations: rotating, flipping, random cropping, zooming, brightness/contrast/saturation adjustments	 <p>The image shows three rows of augmented histology images. Row 'a' shows 'Original' and 'Resized' images at various resolutions (e.g., 1256, 256). Row 'b' shows 'Original' and 'Cropped' images at various coordinates (e.g., 125, 256). Row 'c' shows 'Original Image' and six 'Augmented' images with various transformations like rotation, flipping, and color adjustments.</p>
Feature Extraction: Morphological Operations a. Dilation: Pixel value is max(8 immediate neighbors) b. Erosion: Pixel value is min(8 immediate neighbors) c. Opening: Pixel value is max(min(8 immediate neighbors))	 <p>The image shows a grid of histology images processed with morphological operations. The columns are labeled: 'Positive Sample', 'Gray Dilation', 'Red Dilation', 'Green Dilation', and 'Blue Dilation'. Each column contains a series of images showing the effect of dilation and erosion on the tissue structure, with labels like 'Positive Sample', 'Positive Erosion', 'Negative Sample', and 'Negative Erosion' interspersed.</p>

8 Results

8.1 Machine Learning Models

Experimentation was conducted on the K-nearest neighbors, support vector machine, and mini-batch K-means clustering algorithms in order to determine optimal parameters for the pre-processed data 3. For the gradient boosting classifier, we used RandomizedSearchCV from scikit-learn to find the more-optimal parameters. Then, these parameters were used to fit these algorithms to both pre-processed and raw validation data 4. The conclusion from these algorithms was that the gradient boosting classifier performed the best given the pre-processing steps, but was not run on the raw validation images, which makes it difficult to assess the benefits of the pre-processing steps for this particular model. K-means clustering was performed only on the training split of the data, but with all the corresponding pre-processing steps. Finally, it was seen that both the K-neighbors classifier and support vector machine classifier performed better with the raw validation images, suggesting that our subjective decisions of pre-processing techniques to apply to the images resulted in potential information loss and decrease in performances.

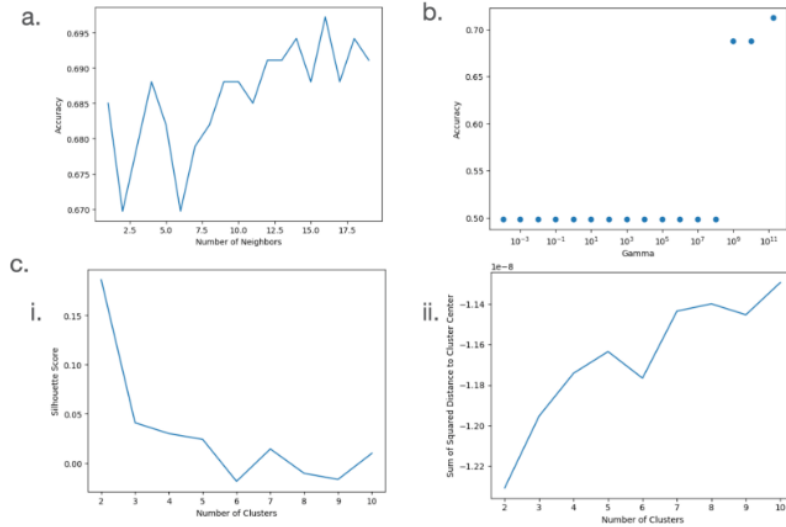


Figure 3: Experimentation on machine learning model parameters for preprocessed data: (a) K-nearest neighbors: Accuracy given number of neighbors. (b) Support vector machine: Accuracy given Gamma parameter value (influence of training sample’s proximity). (c) Mini-batch K-means clustering optimization: (c.I) Silhouette score and (c.II) sum of distances to cluster centers given number of clusters.

Model	Variant	Validation Accuracy
GradientBoostingClassifier (subsample= 0.8, n_estimators=300, min_samples_split=2, min_samples_leaf=4, max_depth=7, learning_rate=0.1)	Reinhard, CLAHE, Pixel Normalization	81%
KNeighborsClassifier (n_neighbors=16)	CLAHE, Macenko, Grayscale, Opened	70% (same as raw)
MiniBatchKMeans (n_clusters = 2)	CLAHE, Macenko, Grayscale, Opened	N/A, but 2 clusters had best performance
svm.SVC (gamma='scale')	CLAHE, Macenko, Grayscale, Opened	70% (raw 73%)

Figure 4: Machine learning model results

8.2 Deep Learning Models

Performance varied across models are listed in tables 3, 4 and 5. We can observe that even the base versions of the models can beat the specialised DenseNet proposed in [11]. We are able to achieve the best test accuracy with the VIT-Base [1] as seen in 5. All the models have been implemented using Pytorch, a deep learning library [6].

9 Discussion

From the above explorations, we have realised a few aspects about the dataset. Preprocessing appears to have loss of information in some cases and decrease the performance in case of some deep learning approaches. As for the machine learning models, gradient boosting seems to have the best impact test accuracy. We also observe that off the shelf deep learning models do a decent job at this seemingly new task. We train on vision models with very different sizes to understand how the number of parameters in the model helps in the task. We observed that preprocessing techniques especially helped in the case of VIT.

Table 3: ResNet-50 results

Model	Variant	Train Acc	Val Acc	Test Acc
Baseline	PCam-DenseNet	-	-	89.8
ResNet 50	Base	91.74 \pm 0.38	89.83 \pm 0.78	88.69 \pm 0.25
	RGB2HED	64.56 \pm 0.4	65.66 \pm 0.22	63.31 \pm 0.52
	Wavelet Transform	91.36 \pm 1.0	86.00 \pm 1.5	86.53 \pm 0.45
	CLAHE	92.51 \pm 1.3	89.82 \pm 0.8	88.49 \pm 0.35
	Reinhard	91.46 \pm 0.8	89.36 \pm 0.22	88.20 \pm 1.2
	Opening	90.53 \pm 0.92	86.84 \pm 0.25	86.82 \pm 0.5
	Macenko	92.14 \pm 0.86	89.75 \pm 1.23	87.97 \pm 0.7

Table 4: UNet results

Model	Variant	Train Acc	Val Acc	Test Acc
Baseline	PCam-DenseNet	-	-	89.8
UNet	Base	95.29 \pm 1.3	90.95 \pm 0.8	90.48 \pm 0.5
	CLAHE	94.58 \pm 1.2	90.55 \pm 0.52	89.83 \pm 0.3
	Reinhard	92.95 \pm 1.7	90.08 \pm 0.33	90.61 \pm 0.42
	Macenko	95.71 \pm 0.95	90.41 \pm 0.34	89.91 \pm 0.72

10 Future Steps

We would like to explore how machine learning techniques can be used as preprocessors to build on. A promising area of exploration is ensembling of machine learning models, and kernelization of the image spaces to boost the test accuracy. For better interpretability of deep learning predictions, we can explore how Shapley values and methods like GRAD-CAM [9] could be helpful to visualize the decision making process.

11 Credits

We are proud of the effort led by our team under the guidance of Prof. Dan Ruan. We are very grateful to her for her guidance throughout the course of this project, and allowing us access to the Hoffman cluster, without which most of the experiments would not have been possible. Each of the tasks were taken up in this fashion:

- Data analysis
 - Dataset distribution, image statistics - Teresa
 - Statistical analysis - Debapriya
 - Dataset distribution - Teresa
- Data preprocessing
 - Noise Reduction, feature extractions - Udbhav
 - Normalization, augmentations - Teresa
 - Stain separation, wavelet transforms - Debapriya
- Machine learning modelling

Table 5: VIT results

Model	Variant	Train Acc	Val Acc	Test Acc
Baseline	PCam-DenseNet	-	-	89.8
VIT	Base	92.02 \pm 2.23	90.47 \pm 0.86	90.86 \pm 0.70
	CLAHE	94.38 \pm 3.2	90.28 \pm 0.77	89.99 \pm 0.90
	Reinhard	92.11 \pm 2.7	90.70 \pm 0.52	90.97 \pm 0.9
	Macenko	92.00 \pm 2.5	91.19 \pm 0.85	92.25 \pm 1.15

- Gradient boosting - Teresa
- KNN, KMeans, SVM - Udbhav
- Deep Learning modelling - Debapriya
- Slide deck preparation - Teresa, Udbhav, Debapriya

We have tried our best to ensure each of us gets to learn something new during the course of this project. Overall, it has been a great learning experience for all of us.

References

- [1] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, Jakob Uszkoreit, and Neil Houlsby. An image is worth 16x16 words: Transformers for image recognition at scale. *CoRR*, abs/2010.11929, 2020.
- [2] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. *CoRR*, abs/1512.03385, 2015.
- [3] Tom Leinster. The euler characteristic of a category, 2006.
- [4] Mahdokht Masaeli, Dewal Gupta, Sean O’Byrne, Henry Tse, Daniel Gossett, Peter Tseng, Andrew Utada, Hea-Jin Jung, Stephen Young, Amander Clark, and Dino Di Carlo. Multiparameter mechanical and morphometric screening of cells. *Scientific Reports*, 6:37863, 12 2016.
- [5] OpenCV. Opencv. <https://opencv.org/>, n.d.
- [6] Adam Paszke, Sam Gross, Soumith Chintala, Gregory Chanan, Edward Yang, Zachary DeVito, Zeming Lin, Alban Desmaison, Luca Antiga, and Adam Lerer. Automatic differentiation in pytorch. 2017.
- [7] F. Pedregosa et. al. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830, 2011.
- [8] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-net: Convolutional networks for biomedical image segmentation. *CoRR*, abs/1505.04597, 2015.
- [9] Ramprasaath R. Selvaraju, Abhishek Das, Ramakrishna Vedantam, Michael Cogswell, Devi Parikh, and Dhruv Batra. Grad-cam: Why did you say that? visual explanations from deep networks via gradient-based localization. *CoRR*, abs/1610.02391, 2016.
- [10] Stéfan van der Walt et. al. scikit-image: image processing in Python. *PeerJ*, 2:e453, 6 2014.
- [11] Bastiaan S. Veeling, Jasper Linmans, Jim Winkens, Taco Cohen, and Max Welling. Rotation equivariant cnns for digital pathology. *CoRR*, abs/1806.03962, 2018.