Segmentation of Glioblastoma Images Using Ensembles of Convolutional Neural Networks

By

Julian Montagut

B.E., McGill University, 2018

Thesis

Submitted in partial fulfillment of the requirements for the
Degree of Master of Science in Biomedical Engineering at Brown University

PROVIDENCE, RHODE ISLAND

MAY 2020

This thesis by Julian Montagut is accepted in its present form
by the Center of Biomedical Engineering as satisfying the
thesis requirement for the degree of Master of Science.

Date 4/21/2020

Dr. Jonghwan Lee, Advisor

Recommended to the Graduate Council

Date 4/20/2020

Dr. Vikas Srivastava, Reader

Date 04/20/2020

Dr. Nicola Neretti, Reader

Approved by the Graduate Council

Date 04/20/2020

Dr. Marissa Gray, Biomedical Engineering Master's Program Director

Preface and Acknowledgements

My foremost gratitude goes to Dr. Jonghwan Lee, my principal investigator and advisor. Dr. Lee provided many opportunities for growth during my graduate studies and offered valuable insight throughout my research. Additionally, I would like to thank Dr. Marissa Gray, the Master's Program Director for Biomedical Engineering, for clearly communicating expectations for Master's candidates and helping to keep me on track during my time at Brown University. Furthermore, I am grateful for Ki-soo Jeong, a Ph.D. candidate in Biomedical Engineering who worked in the Lee Lab. Mr. Jeong helped extensively in the earlier stages of this research by providing the primary histology images examined throughout this thesis and manually delineating the tumor boundary in each. We conducted this research using computation resources and services at the Center for Computation and Visualization, Brown University. Their supercomputer, Oscar, was essential in the running of our model and the collection of data throughout our project. I would also like to thank Ramisa Fariha, my contemporary and fellow Master's candidate in the Lee Lab. Ms. Fariha was consistently available to provide advice and support throughout as an outstanding researcher and a good friend. Finally, I would like to thank my parents, whose invaluable support and guidance during my research and throughout my life were critical in making me the scientist that I am today.

Table of Contents

List of Tables

List of Illustrations

1. Introduction

Cancer presently stands as the second largest cause of death globally[1]. Brain cancers contribute a sizable portion of these fatalities, resulting in 227,000 total deaths in 2016[2]. More disturbingly, incidences of brain cancer have been rising worldwide[2], with new diagnoses being most common in Canada and Northern European countries[3].

Machine learning methods have rapidly emerged as a promising approach to oncological imaging studies[4–6]. In particular, convolutional neural networks (henceforth referred to as CNN's) have rapidly gained traction in this realm[5]. These neural networks employ convolution to detect local changes in pixel values[7]. This process is presented in visual form in Figure 1 below[4]:

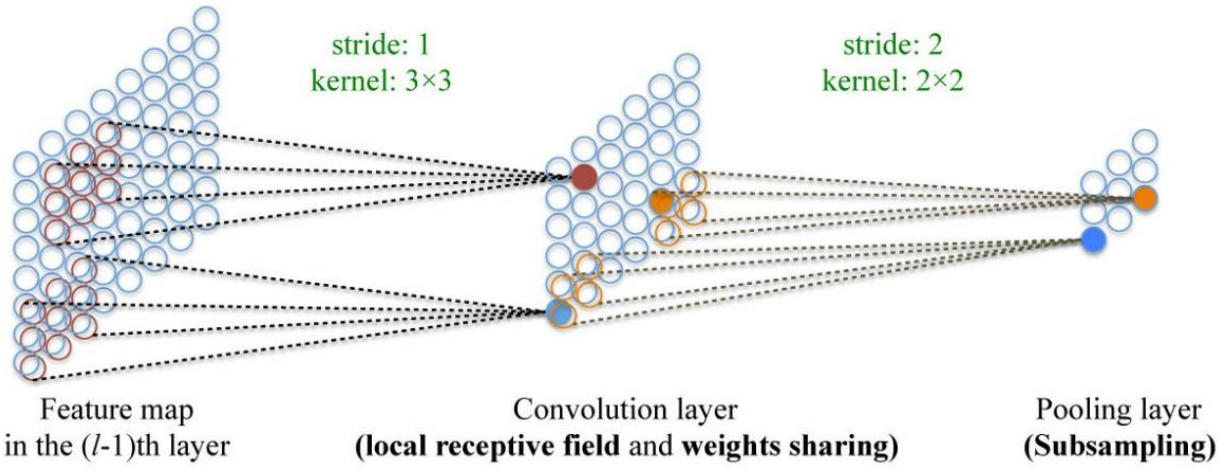


Figure 1: Visual Illustration of the Effects of Convolution[4]

Among biomedical imaging studies, image classification and image segmentation have emerged as the two most dominant categories[8]. In the former case, the system in question examines

a biomedical image and returns a single prediction, such as cancerous or non-cancerous[8]. Likewise, image segmentation seeks to separate an image into several distinct regions[9]. The latter is of particular interest for brain cancer, as it can allow for the localization of tumors for treatment or resection[10].

Various imaging modalities are available for brain tumor analyses using CNN's[10]. The most prevalent of these by far is magnetic resonance imaging (MRI)[11–15]. In addition to its non-invasive nature, MRI offers superior contrast and safety over its counterparts[10]. Regardless, for initial diagnosis, the reliability of biopsy has yet to be surpassed, as experts have consistently referred to it as a gold standard[16,17].

Historically, these visualization approaches have occupied distinct niches. Biopsy was typically the preferred choice for brain tumor detection (image classification)[16,17], whereas scientists and medical professionals typically employed MRI to delineate brain tumor boundaries (image segmentation)[18,19]; however, a study by *Li et al.* in 2014 demonstrated the capabilities of CNN's for examining brain tumor images of various modalities, namely MRI and positron emission tomography (PET)[20]. Furthermore, recent research published this year by *Hollon et al.* demonstrated a successful implementation of CNN's for segmentation of brain tumor images obtained by simulated Raman histology (SRH)[21], a method which produces rather similar results to conventional biopsy staining[22]. For these reasons, we believe that it is worthwhile to study the segmentation of stained histopathological images of brain tumors using CNN's.

When assessing the behavior of machine learning models, the Receiver Operating Characteristic (ROC) curve has consistently shown success[23]. While these curves themselves can provide valuable information, studies frequently assess them in terms of the associated areas under

them[24]. (often denoted as AUROC[25], but henceforth referred to as AUC) Indeed, several histopathological image studies utilizing CNN's made use of this metric[25–28].

However, a promising counterpart to the ROC curve is the precision-recall curve[29]. Rooted in information retrieval systems[30], this curve offers superior insight for datasets whose constituent classes are imbalanced[31]. This is of particular interest for biomedical image studies, as class imbalance remains a pervasive issue in this field[32]. A sample case comparing two systems in ROC and precision-recall space is presented in Figure 2 below[29]:

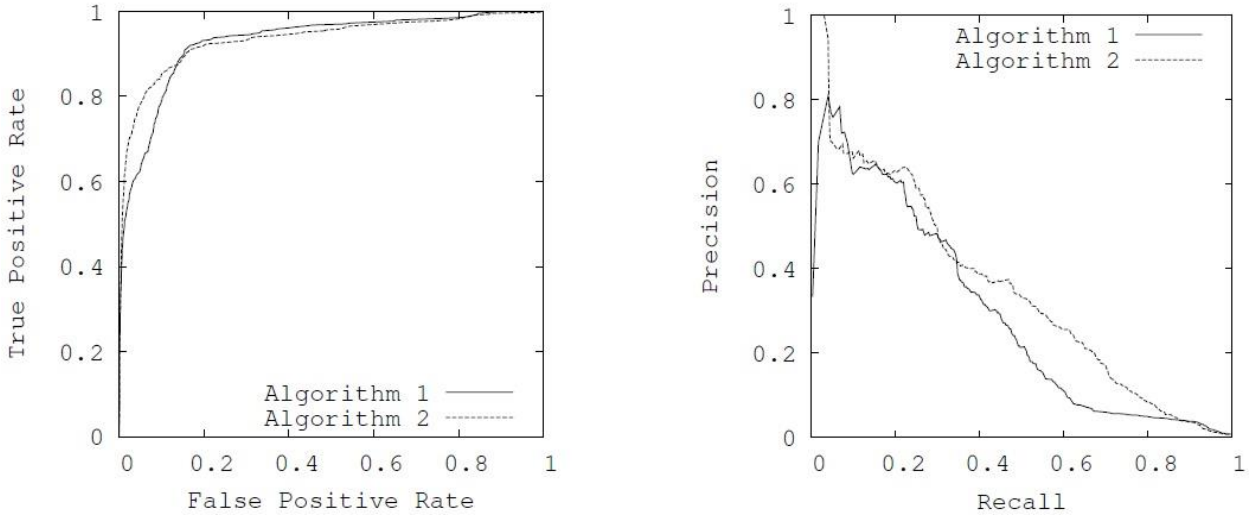


Figure 2: Comparison Between ROC and Precision-Recall Curves[29]

Furthermore, much like a ROC curve, a precision-recall curve can also be quantified by the area beneath it, conventionally termed the average precision (AP)[33]. An associated metric is the mean average precision (MAP)[34], which represents the mean of several AP values for various classes[35]; however, in practice, data scientists often use these terms interchangeably[36]. Thus, we will refer to this property as MAP for the remainder of this thesis.

Despite its apparent advantages for skewed datasets, the precision-recall curve has consistently been neglected in literature in favor of the ROC curve[31]. This is of further interest to us, due to the aforementioned class imbalance widespread in biomedical imaging analyses[32]. Therefore, we believe that it is worth studying and evaluating precision-recall curves and MAP values in the context of a biomedical image segmentation project.

The purpose of this thesis was to carry out image segmentation on stained histopathological images of brain tumors using CNN's. We utilized a patch-based approach, in which we classified smaller image patches as tumor or normal, rather than a semantic (i.e. pixel-wise) segmentation method for the sake of simplicity. Likewise, we quantified the performance of our classifier using both AUC and MAP to determine the validity of using the latter metric in a biomedical image segmentation environment.

2. Background

The contemporary CNN has its roots in the neocognitron architecture introduced by *Fukushima* in 1980[37]. This primitive archetype employed circular convolutional kernels[37] and failed to utilize a standardized form of training[38], such as stochastic gradient descent[39]. Despite this, it achieved moderate success on basic studies in these early days of artificial intelligence[40,41].

The first instance of the use of CNN's for biomedical image analysis occurred in 1995[42]. In this study, *Lo et al.* developed a system using CNN's to detect cancerous lung nodules in digitized radiograph images of cancer patients[42]. Such early applications of neural nets to biomedical images typically modeled their workflows after the steps radiologists would take during diagnoses[42,43].

The following years saw limited use of CNN's for biomedical image segmentation applications. In 2005, *Ning et al.* produced a pipeline utilizing CNN's for locating cellular components in differential interference contrast (DIC) videos of developing *C. elegans* embryos[44]. Similarly, *Turaga et al.* examined scanning electron microscopy (SEM) images of retinal tissues using CNN's in 2010[45].

However, in 2012, CNN's received a resurgence of interest as a result of the 2012 ImageNet Large Scale Visual Recognition Challenge (ILSVRC2012)[46]. At this competition, CNN's demonstrated excellent performance and far surpassed comparable machine learning approaches[47]. Namely, *Krizhevsky, Sutskever & Hinton* introduced the AlexNet architecture, which offered unprecedented capabilities at the time[47].

The first major application of CNN's for biomedical image segmentation after this breakthrough occurred in 2013[48]. *Cireşan et al.* utilized this method to detect mitosis in breast

cancer histology images[48]. The following year, *Cruz-Roa et al.* performed patch-based segmentation of breast cancer tissue issues using a CNN in conjunction with a Random Forest[49].

The next landmark development in this field came with the introduction of the U-net by *Ronneberger, Fischer & Brox* in 2015[50]. Based on a previous fully-convolutional CNN architecture for semantic segmentation proposed by *Long, Shelhamer & Darrel*[51], the U-net was developed specifically for biomedical image segmentation[50]. This network performed semantic segmentation by carrying out convolutional down-sampling followed by corresponding up-sampling[50]. As a result, the model architecture adopted a unique U-shape[50], as shown in Figure 3 below[52].

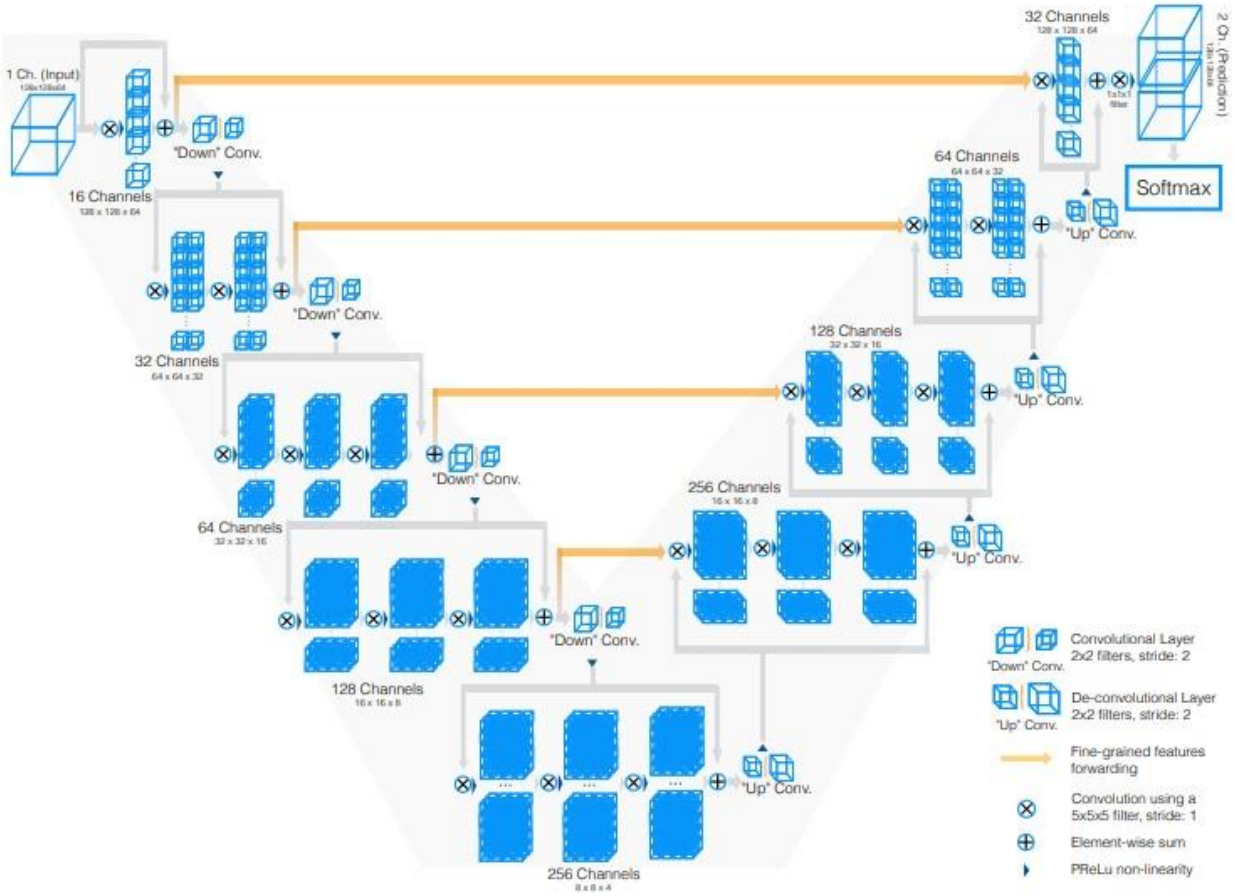


Figure 3: Diagram of the U-net Architecture[52]

Since its introduction, U-nets have rapidly gained prominence for semantic segmentation of biomedical images[53,54]. As recently as 2018, *Shvets et al.* utilized this architecture for robot-assisted surgery[55]. Likewise, in 2016, *Çiçek et al.* produced a modified U-net architecture for three-dimensional data[56].

Despite the aforementioned ubiquity of U-nets, conventional CNN's have continued to see use in histopathological image segmentation studies. *Litjens et al.* published their research on the segmentation of prostate cancer and sentinel lymph node histology images using CNN's in 2016[27]. The same year, *Wang et al.* utilized CNN's to achieve strong results on both classification and segmentation studies of breast cancer images[26]. Later, in 2017, *Liu et al.* further built on the aforementioned results[57].

As previously stated, studies in literature have consistently omitted precision-recall curves and MAP in favor of ROC curves and AUC respectively[31]. While this has generally applied to histological image studies utilizing CNN's as well, this metric has seen scattered use. In 2019, *Mercan et al.* utilized a CNN to segment breast cancer histopathology images and reported their final results in terms of MAP[58]. The same year, *Lu & Daigle* made use of MAP when applying CNN's for studying histology images of liver cancer[59]. Finally, *Schaer et al.* developed an image retrieval system for histopathological images using a CNN, where they quantified performance in terms of MAP and precision-recall curves[60], likely due to their origins in information retrieval[30].

3. Materials and Methods

3.1. Image Preprocessing

This segmentation study examined four primary histopathology (H&E-stained) images of glioblastoma xenograft in mice. This data had been obtained by Ki-Soo Jeong, a graduate student in the lab. The H&E staining is a gold standard in the histopathology of cancer tissue[61]. These four primary images can be observed in Figure 4 below.

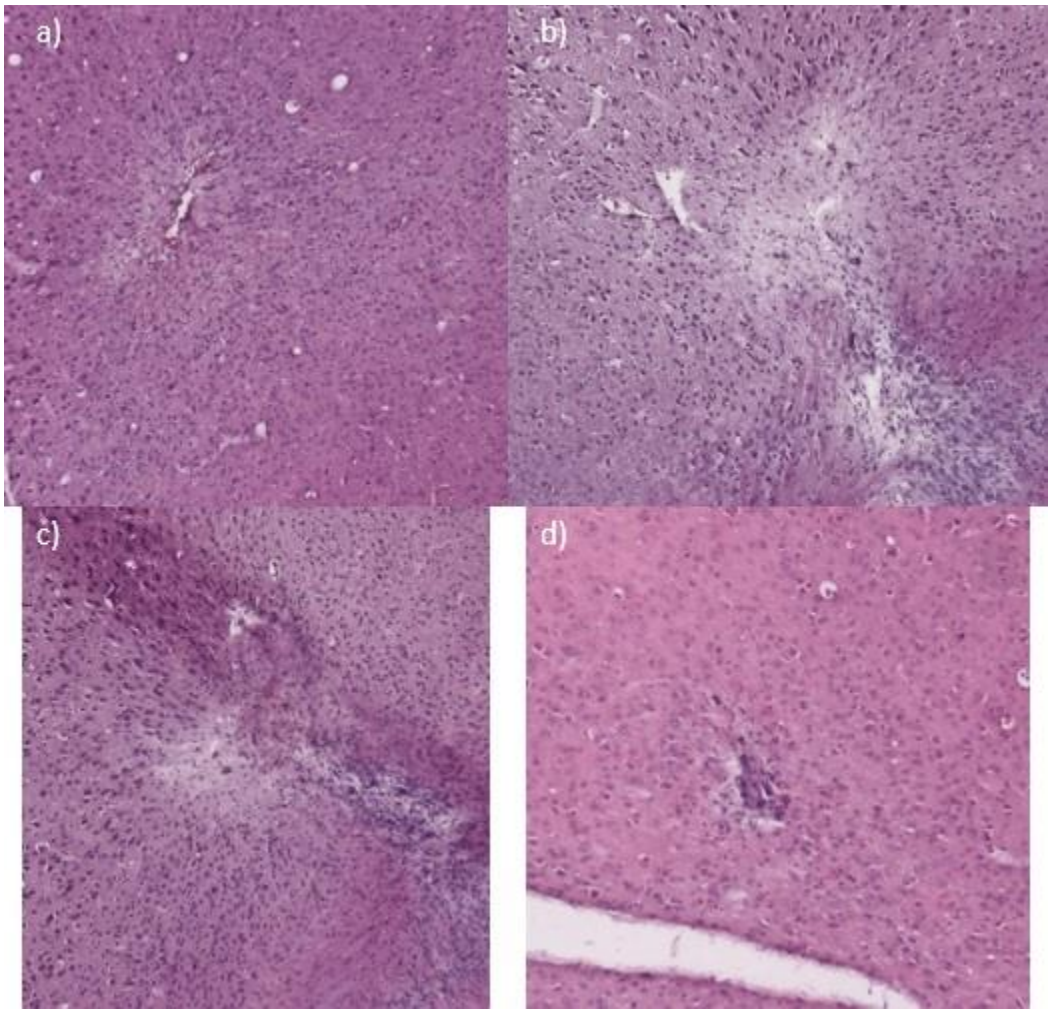


Figure 4: Primary Histopathology Images in this Study, Labeled as Follows: a) 002, b) 003, c) 004, d) 166

We printed these images and provided them to Mr. Jeong, who then traced the tumor in each image. We then opened the corresponding file in MATLAB, recreated the drawn tumor boundary using the roipoly() operation[62], and saved the resulting segmentations in case further examination or manipulation was required.

From here, we divided each image into smaller patches. Furthermore, we also assigned binary labels to each patch based on the criterion that more than half of its pixels resided within the aforementioned tumor boundary and wrote the results to corresponding text files. We initially produced patches of size 30 x 30, 60 x 60 and 120 x 120 in this manner to allow for more directions in the earlier stages of this project. The patch sizes themselves were arbitrary selections based on the general sizes of the primary images; however, as previously stated, the limited number of training samples is a pervasive problem in similar studies[6,8]. Thus, we decided to proceed with the smallest of these patch sizes (i.e. 30 x 30) in order to maximize the number of samples available to our model.

In addition to the patch sizes, we also divided each image based on an overlapping parameter. This Boolean variable indicated whether adjacent patches should overlap by half. (e.g. 15 pixels for 30 x 30 patches) Setting this variable to true (1) both aided in generating more patches for the model and resulted in smoother classification mappings in the later stages of this project. For these reasons, we opted to proceed with the patches obtained while the overlapping setting was on.

Data augmentation is often critical for such studies to offset the limited number of available samples[4,27,48,50]. Therefore, we developed the following five augmentations for each patch produced:

1) Flipping vertically
2) Flipping horizontally

3) Rotating 90°
4) Rotating 180°
5) Rotating 270°

Our goal with these data augmentations was to provide additional patches to our model for training and validation, as well as to make it more robust to the patch orientation. Furthermore, we developed two methods of applying these augmentations. In the former method, we selected a random transformation from those listed above for each patch, applied it and saved the resulting sample. This produced one new patch for each original sample, increasing the total number of patches by a factor of two. Conversely, the latter method applied all of the aforementioned transformations to each patch and saved the results, thereby generating five new samples for each patch and multiplying the total number of patches by a factor of six. Since the latter method entailed more samples available to the model, we chose this option and proceeded accordingly. Finally, it is critical to note that, in both methods, we assigned each patch produced from data augmentation the same label as its original counterpart.

Table 1 on the following page summarizes the results of this image preprocessing stage. In total, we generated nearly 100,000 patches on which we could train and evaluate our model; however, it is critical to note the exceptional nature of Image 166. First, Image 166 provided approximately 53,000 (i.e. ~55%) of these patches, far more than any of its counterparts, due to its much larger size. Furthermore, as shown in subfigure d) of Figure 4, this image contained a unique gash in the lower-left region. These properties were the result of Image 166 being obtained at a later point in time than its counterparts. For these reasons, we sought to quantify the effects of using patches from this image and ascertain that it did not negatively impact the model behavior in our experimental design.

Table 1: Table of Relevant Metrics for All Primary Images

Image ID	Image Dimensions	Mapping Dimensions	Number of Primary Patches	Number of Patches after Data Augmentation
002	808 x 807	52 x 52	2,704	16,224
003	735 x 682	48 x 44	2,112	12,672
004	735 x 787	48 x 51	2,448	14,688
166	1425 x 1424	94 x 94	8,836	53,016
			16,100	96,600

3.2. Deep Learning Model

3.2.1. Model Architecture

When selecting a CNN architecture for this study, our main priority was one which could be adapted for a small patch size. Furthermore, we also desired an architecture whose powerful capabilities had clearly been demonstrated in literature. For these reasons, we selected the AlexNet architecture[47]. As previously stated, this landmark CNN was the winner of the ILSVRC2012, where it offered vastly superior performance to its contemporaries[47]. Since then, experts have employed this architecture in various biomedical imaging studies, due to its powerful capabilities and relative simplicity[26,51,63,64]. A diagram of this architecture is presented in Figure 5 on the following page[64].

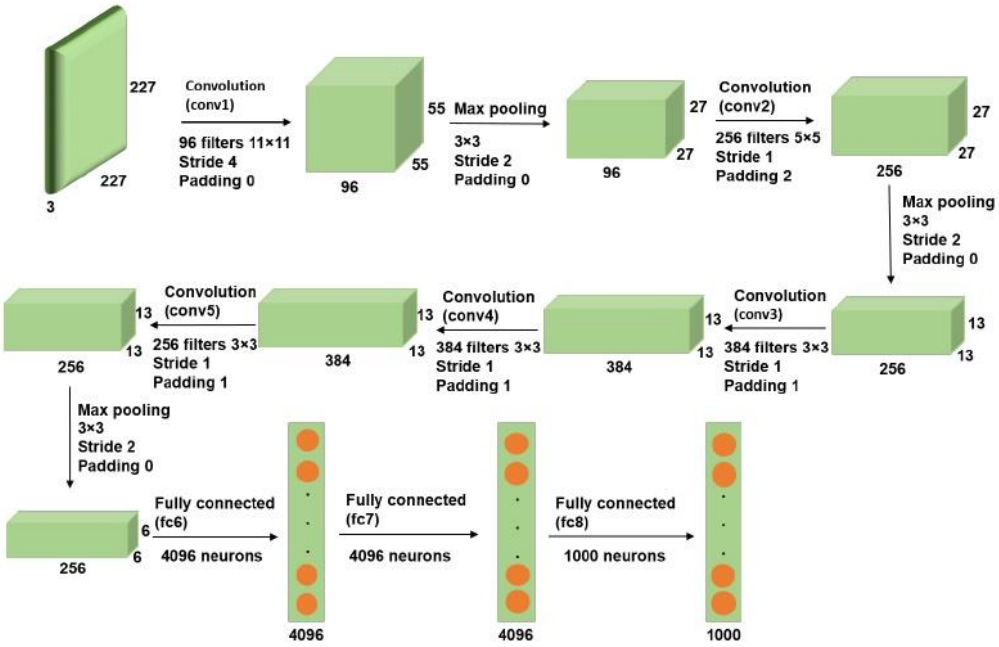


Figure 5: Diagram of the AlexNet Architecture[64]

The AlexNet architecture was originally designed for three-channel inputs of size 224 x 224[47]. Furthermore, due to the three fully-connected layers at the end, this input size is not flexible.[51] Thus, we were unable to implement this architecture on our 30 x 30 patches without some necessary alterations. A diagram of this modified AlexNet architecture is presented in Figure 6 on the next page.

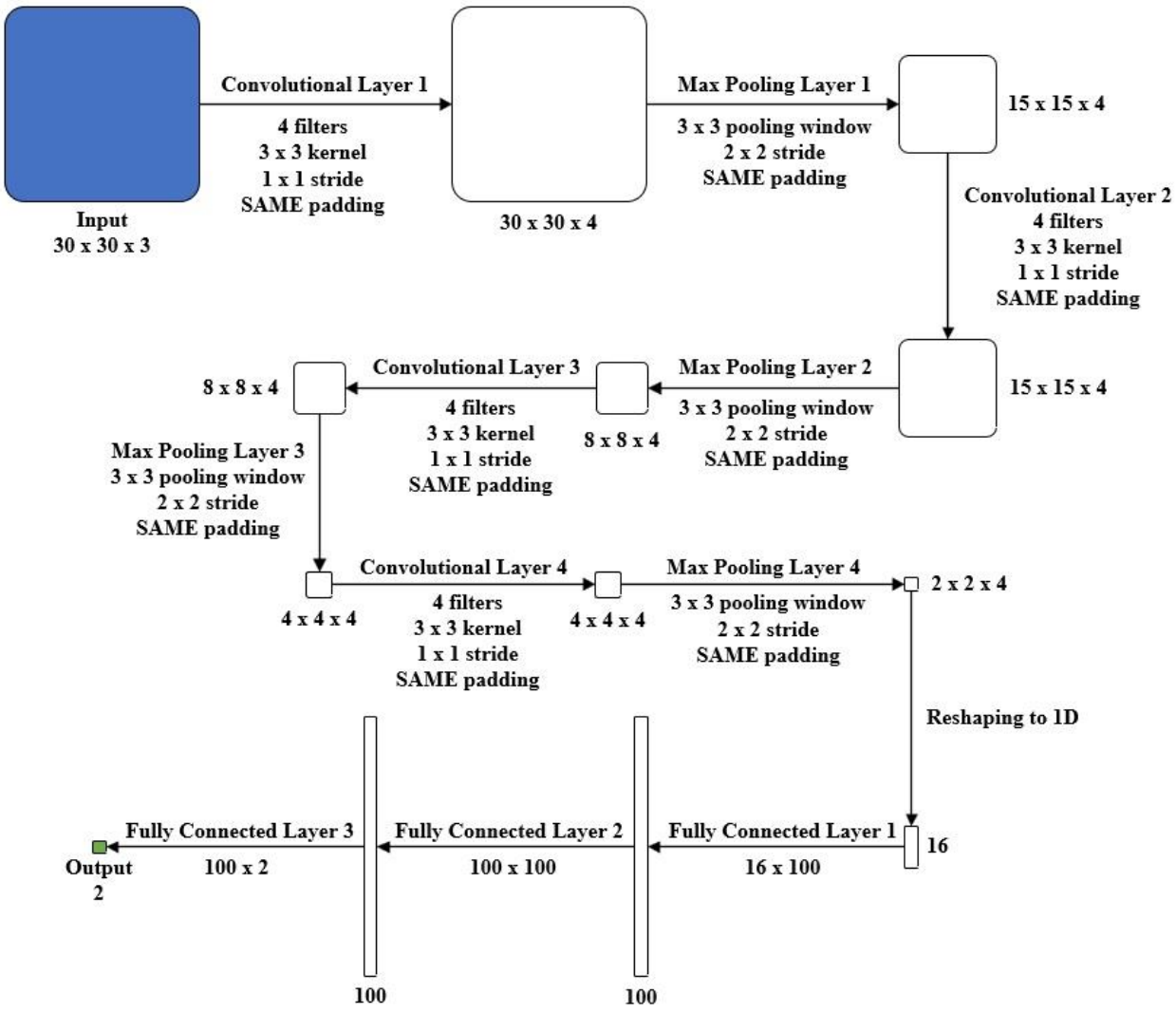


Figure 6: Diagram of the Modified AlexNet Architecture Employed in this Study

The most obvious distinction between this model and its inspiration is the removal of the final convolutional and max pooling layers[47]. We opted to omit these model components, since these would have entailed only a single input per filter to the fully-connected layers (1 x 1 x 4), which likely would have inhibited the predictive capabilities of the system. Similarly, while the original AlexNet architecture includes variable numbers of filters for the convolutional layers on the scale of hundreds[47], our model consistently applied four filters at each such layer. This choice came after we attempted to increase this number to 64 at an early stage in the project and observed sizable

difficulties on the validation set. Finally, we constructed the model to return outputs from fully-connected layers of size 100. This was an arbitrary selection that better suited the reduced scale of both the output from the final max pooling layer and the adjusted model as a whole.

As previously stated, our model initially contained four convolutional and max-pooling layers in series. The convolutional layers utilized stride lengths of 1 x 1 alongside SAME padding, so as to prevent dimensional reduction. Conversely, the max pooling layers involved stride lengths of 2 x 2 to ensure down-sampling. We specified the filter kernels and pooling windows in the convolutional and max pooling layers respectively to be of size 3 x 3 to better fit the aforementioned smaller scale of the model. We utilized ReLU as an activation function after each convolutional layer and the first fully-connected layer, largely due to its pervasiveness in deep learning[38] and proven superior performance[65,66]; however, we opted to utilize sigmoid after the second fully-connected layer based on evidence that a bounded activation function at the end of the model could allow for more rapid training[67]. It is also worth noting that we examined the effects of replacing ReLU with Leaky ReLU[68] at an early stage, but could not identify explicit improvement on the validation set and therefore chose to continue using ReLU. Finally, the last fully-connected layer returned one logit for each image class (i.e. normal or tumor), which were subsequently converted to probabilities using softmax[69].

We further altered our model to combat overfitting. We performed batch normalization[70] on the output from each convolutional layer for these reasons. Furthermore, we configured the model to apply dropout[71] prior to the first two fully-connected layers during training. The dropout probability for these purposes was 0.2, which we based on a recent study[72]. It is worth noting that such probabilities are typically set to 0.5[14,71,73]; however, since we supplemented this method with batch normalization, we continued to use a lower probability.

3.2.2. Model Implementation

We constructed the aforementioned deep learning model in Python 3.6.6 using TensorFlow 1.11.0[74]. After preparing code for these purposes, we uploaded the files to Oscar, the university supercomputer provided by the Center for Computation and Visualization (CCV), and submitted jobs using bash scripts. We requested two cores with 16 gigabytes of memory and 12 hours of runtime for each job, which we adapted from sample scripts provided by the CCV.

Prior to training and evaluating the model, it was critical to construct the training, validation and testing sets. We used the training set to train the model and the validation set to periodically assess system performance, as well as roughly tune various hyperparameters at early stages of the project. Likewise, we obtained predictions on the testing set in the form of softmax probabilities after training had concluded. The exact patches designated for each set varied based on the study in question, which we discuss in later sections.

We randomly divided image patches specified for the training and validation sets between these sets with a 9:1 split. This proportion was based on literature[72] and was relatively skewed for these purposes[26–28]; however, we came to realize that employing the training set as was strongly inhibited the learning capabilities of our model, due to the strong bias towards normal patches. (~86%) This is a rather common problem in similar studies and is typically remedied by enforcing class balance in the training set[44,57,75]. Thus, we supplemented this division between the training and validation sets with a training set balancing step to produce approximate balance between normal and tumor patches in the training set. More specifically, we selectively removed normal samples from the training set at a rate calculated from the numbers of tumor and normal patches.

We further quantified the effects of this step based on several metrics, which we discuss in Section 4.1.2.

At this point, we also opted to identify all trainable parameters in our architecture and print their names to the job output files. We present and discuss these results in Section 4.1.1. After this step, we determined the names of all patches in the training and validation sets, as well as the corresponding labels, and assigned them to respective arrays. We then trained the model on batches of training patches, while periodically evaluating on batches of validation samples and writing the results to comma-separated values (CSV) files for easier handling.

We selected a batch size of 1024 for training and validation. Based on the manner in which we structured our code, certain proportions of the training and validation sets were omitted during each iteration; however, by performing a large number of training epochs and repeatedly evaluating on the validation set, we were able to mitigate this issue. We based our choice of batch size on early analyses which determined that 1024 limited the omitted portions to reasonable levels. A visual representation of the aforementioned issue can be observed in Figure 7 below.

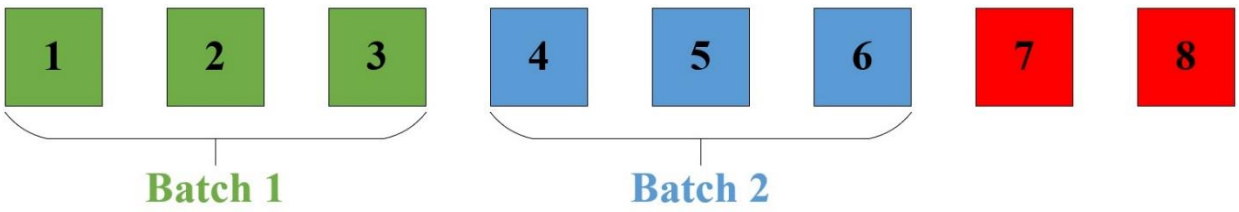


Figure 7: Sample Case Illustrating the Omission of Samples for a Set Size of 8 and a Batch Size of 3. Since 8 is not evenly divisible by 3, the model does not encounter the remaining 2 patches.

After reading in the training patches from their respective file names, our model performed the forward pass to obtain predictions and subsequently trained the model. Our choice of loss function was cross-entropy loss, due to its ubiquity for binary classification tasks in deep learning[76,77].

Likewise, we selected the Adam optimizer based on its established advantages over stochastic gradient descent[78].

We configured our model to evaluate itself on the validation set immediately before training and after every 10 epochs to assess the initial state of the model and monitor performance respectively. This number of epochs was an arbitrary selection which allowed us to sufficiently understand the model behavior. During validation, our model read patches from the validation set and obtained predictions; however, unlike during training, the model than converted the softmax probabilities to binary outputs (normal or tumor) and compared the outputs to the corresponding labels. Our model selected the class whose probability was higher as the predicted class of the patch in question, effectively applying a probability threshold of 0.5. We selected this criterion, as it seemed to be a logical one. After each pass over the validation set, our code wrote several important metrics to the aforementioned CSV file. The most important of these were the sensitivity, specificity and overall accuracy; however, a full list of metrics can be seen in Appendix 7.1.

We initially used the results on the validation set to loosely tune the model hyperparameters. Most notably, we opted to perform 250 training epochs for each run, since we observed relative stability in the accuracy, sensitivity and specificity values on the validation set by this point. We also obtained a general range of learning rate values which seemed to entail reasonable performance; however, we did not decide on a single learning rate value until our preliminary study, which we discuss in Section 3.3.

Upon reaching the final training epoch, we configured our code to obtain time values immediately before and after training over one batch of samples. We then computed the difference and wrote the results to an associated text file, such that we could later examine the model

performance. We likewise repeated this analysis on the validation set after training had concluded to assess this behavior as well. We present these results in Section 4.1.1.

Finally, we evaluated our model on the testing set. Unlike the validation set, we did not apply an effective threshold to the outputs and rather obtained the softmax probabilities for both classes. We then wrote these probabilities to corresponding text files for further analysis. It is also worth noting that it was critical to obtain a prediction for each testing patch to reconstruct classification mappings. Therefore, we developed a solution to the aforementioned omission of some patches during training and validation for the testing set. More specifically, we structured our code to obtain predictions on batches which partially overlapped with previous ones, while saving the results for patches only present in the former. A diagram of this process can be observed in Figure 8 below.

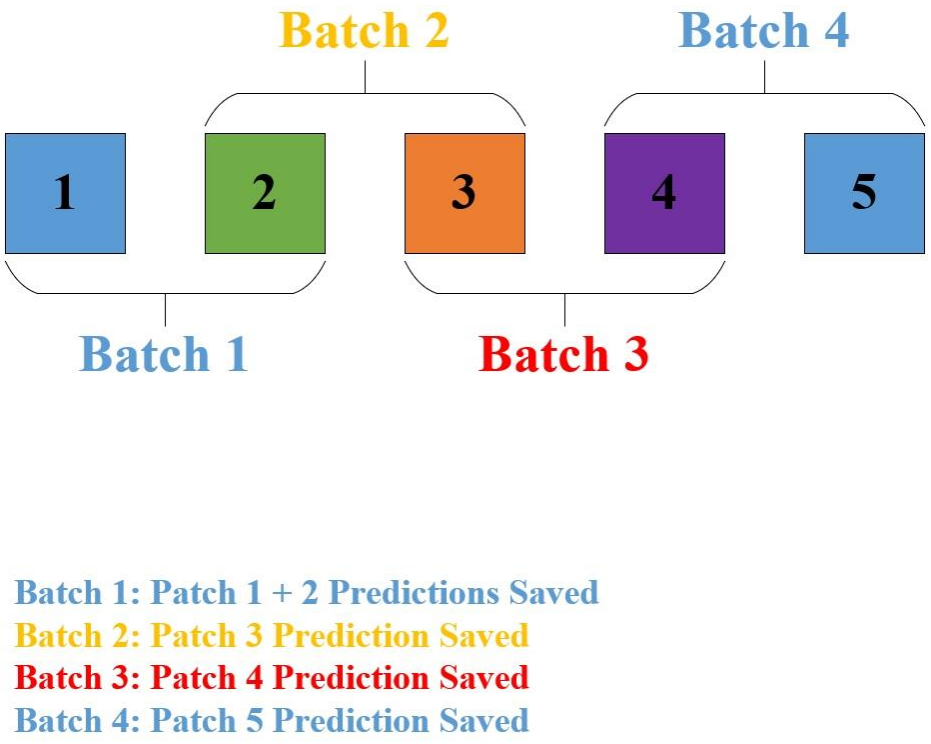


Figure 8: Sample Case Illustrating the Solution to the Batch Size Issue on the Testing Set. After saving all of the results from the first batch, the subsequent batches only shift by one sample and append the result of the new patch.

3.3. Preliminary Study

Our first venture involving the testing set was a preliminary study. Here, we constructed the training and validation sets from all available patches (both original and from data augmentation), while utilizing the original patches for each image as the testing set. Our code wrote the softmax probabilities of all patches in the testing set to corresponding text files, such that we could reconstruct probability heatmaps and compute key classifier metric values (AUC and MAP). A diagram of this setup is presented in Figure 9 below.

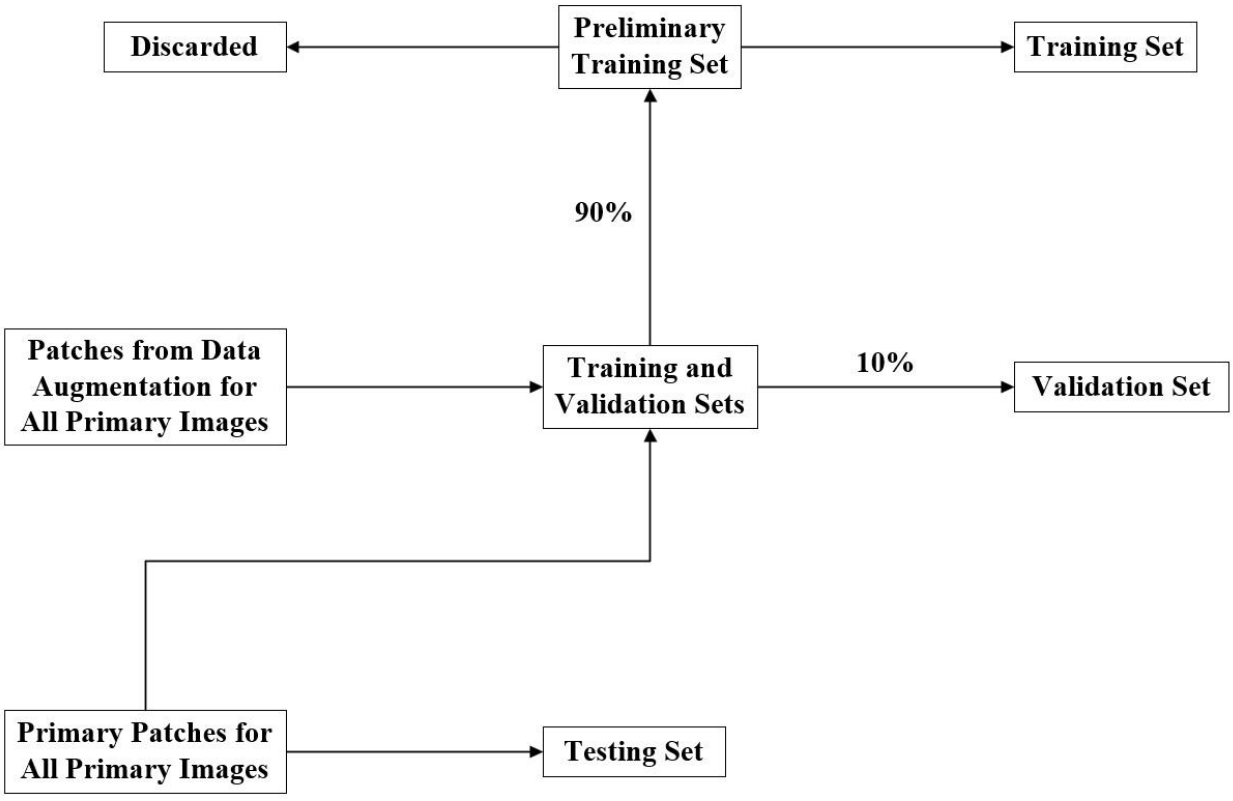


Figure 9: Flowchart of the Construction of the Training, Validation and Testing Sets in the Preliminary Study

We performed ten runs of this process for each learning rate evaluated. This number was an arbitrary selection which we chose to mitigate variability between runs caused by a number of factors. (e.g. different training set compositions) Likewise, we examined five distinct learning rates at this stage: 0.01, 0.001, 0.0005, 0.0001 and 0.00005. This selection consisted of both learning rates which we had confirmed to perform well on the validation set, as well as additional options for the sake of thoroughness. We also attempted to refine the learning rate further, but were unable to obtain statistically significant results, as we discuss in Section 4.2.

It is worth noting that the model encountered the original patches in both the training and testing sets and therefore trained on a portion of the testing set. In typical machine learning analyses, this would be considered unacceptable, as it has been documented to artificially exaggerate the predictive capabilities of the model[79]; however, we performed this analysis for two reasons. The first of these was to confirm that the use of a large number of patches from Image 166 did not skew the results and the aforementioned gash discussed in Section 3.1 did not impede model performance. Likewise, this initial analysis allowed us to identify an optimal learning rate for each key classifier metric, thereby fixing one tunable parameter and preventing complications in future work.

After performing the aforementioned runs, we transferred the text files from our Oscar account to a local MATLAB server for further examination. Namely, we read in the predictions and associated labels, reconstructed a probability heatmap from each set of probabilities and applied a moving threshold to convert the softmax values to binary predictions. This threshold began at zero, essentially predicting that all patches were tumors, and iteratively increased by 0.01 until reaching one, where all patches were predicted to be normals. We chose this threshold step size arbitrarily as a small value to allow for smooth ROC and precision-recall curves.

Our code then compared the binary predictions for each threshold value to the corresponding labels in order to calculate the proper metrics. (true positive rate and false positive rate for ROC, precision and recall for precision-recall[31]) Computing these metrics for each threshold value allowed us to trace ROC and precision-recall curves for each probability heatmap. Finally, we were able to calculate AUC and MAP values by numerically calculating the integrals of these curves. Our choice for numerical integration was the Trapezoidal Method, since the formula for Simpson's Rule with unevenly-spaced data is rather complicated[80]. After obtaining AUC and MAP values for all probability heatmaps at each learning rate, our code wrote these values to associated CSV files. It is also worth noting that increasing the threshold eventually produced classification mappings with zero true positives and false positives, causing the denominator of the precision equation to be equal to zero[31]. Thus, when applying the moving threshold for the precision-recall curves, we added a conditional statement to check if the aforementioned condition had occurred and to break if it returned true.

Once our MATLAB code had concluded, we transferred the AUC and MAP values from the aforementioned CSV files to Excel files for more careful examination. We selected optimal learning rates for both metrics based on their relative performances across all primary images before moving on to the following study. We also performed analyses of variance (ANOVA's) for both AUC and MAP over each primary image followed by Tukey tests wherever suitable. We performed these statistical tests, as well as all such tests in this thesis, using R version 3.5.2. Likewise, all statistical tests discussed here had a level of significance of 5%, due to its prevalence throughout statistics[81]. Finally, we plotted representative probability heatmaps for the aforementioned learning rates to qualitatively assess the behavior of the model at this stage.

3.4. Main Study

3.4.1. Heatmap Averaging

We followed our preliminary analysis with a main study, in which the testing set did not overlap with the training and validation sets at all. More specifically, we configured our code to designate original patches from a single primary image as the testing set, while constructing the training and validation sets using patches (both original and from data augmentation) from all remaining images. We present this process in the form of a flowchart with Image 002 as the testing set in Figure 10 below.

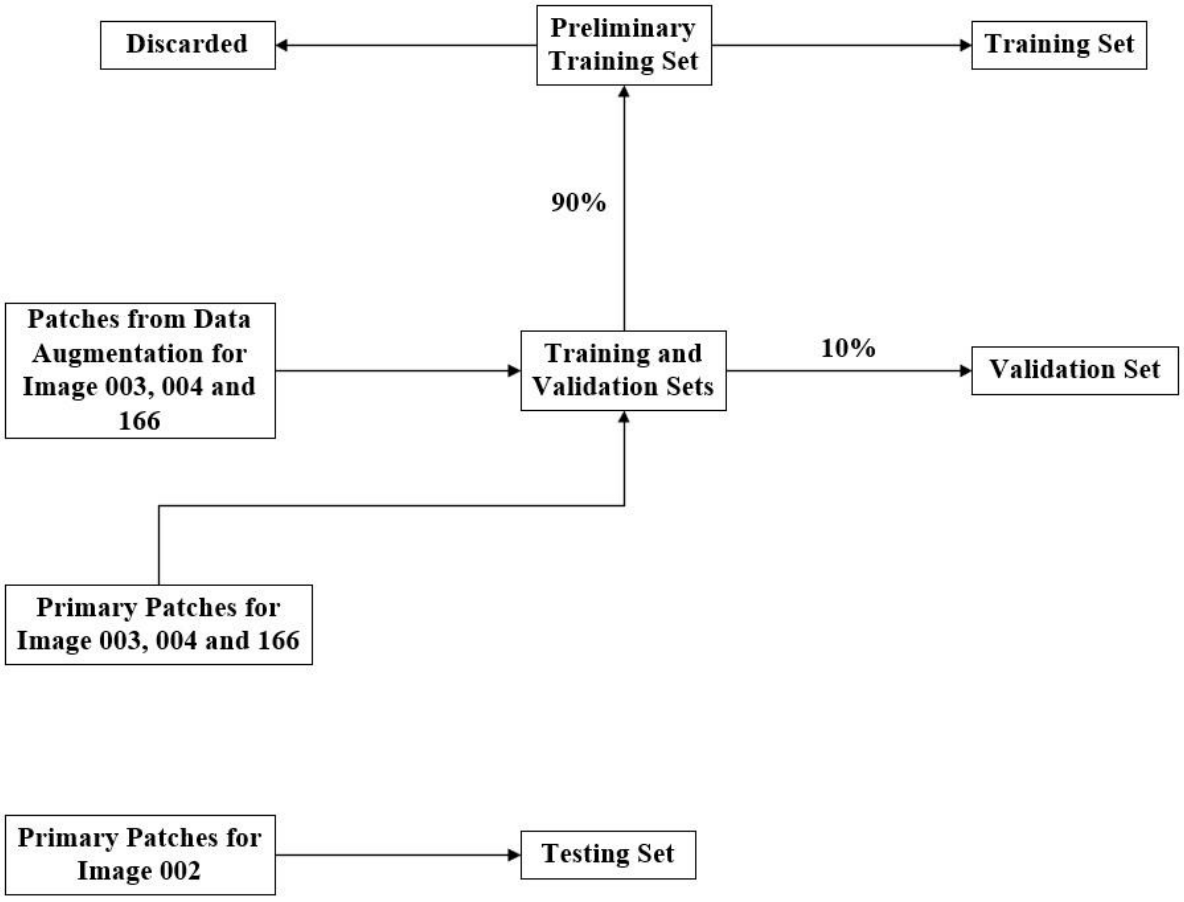


Figure 10: Flowchart of the Construction of the Training, Validation and Testing Sets in the Main Study with Image 002 Designated as the Testing Set. Note the complete separation between the testing set and the training and validation sets.

We performed runs at the optimal learning rates for both key classifier metrics, while designating Image 002, 003 and 004 as the testing sets. We decided to omit Image 166 as a testing set image, since it contributed significantly more patches than its counterparts and would therefore deprive the model of copious training set data. Additionally, we opted to perform ten runs for each combination of learning rate and testing set image, both to maintain consistency with the preliminary study and for the reasons discussed in the previous section. In total, this resulted in ten runs for each of the three testing set images, or thirty total runs per learning rate.

We calculated the proper key classifier metric values for all runs of each learning rate in the same manner as before; however, we also reduced the threshold step size to 0.001 at this stage. This was largely to ensure that the precision-recall curves produced at the end of this study did not appear jagged. It is also worth noting that quick calculations confirmed that this alteration changed AUC and MAP values by approximately 0.2%, so we considered any effects on these values incurred by changing the step size to be negligible.

At this stage, we also produced additional heatmaps via heatmap averaging. As indicated by its name, we summed every probability heatmap available for a given test image at a specific learning rate and divided each element by the number of heatmaps. (i.e. ten) Experts have consistently documented the capabilities of such ensemble learning processes in literature[25,48,82–84]. Therefore, we carried out this heatmap averaging and computed corresponding key classifier metrics for the novel heatmaps. Just as before, our code wrote all of the aforementioned AUC and MAP values to associated CSV files.

In addition to further examining the results of this stage in Excel, much like the preliminary study, we also carried out a series of statistical tests. First, for both learning rates, we carried out a t-test comparing the key classifier metric values for primary probability heatmaps to that of the

averaged heatmap for each testing image. We supplemented this with a paired t-test between the three mean key classifier metric values before heatmap averaging and the corresponding values obtained after heatmap averaging for each learning rate.

Finally, we determined the optimal threshold for each probability heatmap, recorded them to associated CSV files and further examined them in Excel. Our criteria for determining these thresholds was those which minimized the distance from ideal behavior in the corresponding space. This ideal behavior differed between ROC and precision-recall curves. In the former case, an ideal classifier would offer a true positive rate and a false positive rate of one and zero respectively[85]. Thus, the equation for the distance as applied to ROC curves is as follows.

$$d_{ROC} = \sqrt{(1 - TPR)^2 + FPR^2} \quad (1)$$

In the above equation, TPR and FPR denote the true positive rate and the false positive rate respectively. Conversely, an ideal classifier in the latter case would offer both a precision and a recall of one[29]. Therefore, the equation for the distance as applied to precision-recall curves is shown below.

$$d_{P-R} = \sqrt{(1 - P)^2 + (1 - R)^2} \quad (2)$$

Just as before, P and R in the aforementioned equation designate the precision and the recall respectively. We also applied these optimal thresholds to both representative original probability heatmaps and those obtained from heatmap averaging to generate corresponding classification mappings, which we then saved to compare at a later stage.

3.4.2. Median and Gaussian Filtering

After quantifying the effects of heatmap averaging, we applied supplemental filtering to the aforementioned probability heatmaps in this study. Namely, we introduced two additional filtering steps in series. The former of these was median filtering, which changed each patch to the median of its neighbors designated by a kernel in order to remove noise[86–88]. Gaussian blurring (henceforth referred to as Gaussian filtering) followed this step, which entailed both the removal of additional noise and the blurring of the heatmap overall[89,90]. We used the medfilt2()[91] and imgaussfilt()[92] operations in MATLAB for these purposes respectively. The combined effect of these steps was to fully separate the tumor and normal regions in each probability heatmap upon the application of the optimal threshold, thereby delineating the tumor boundary.

Each of these filtering processes contained one tunable parameter, namely the kernel size and the filtering standard deviation for median filtering and Gaussian filtering respectively. We sought to identify the optimal set of filtering parameters for each key classifier metric and apply it generally over every testing image for the final parts of this study, rather than filtering differently for each image. Therefore, we repeated the previous analysis, while applying median and Gaussian filtering of various degrees to each probability heatmap before calculating the new key classifier metric values. We also performed the same heatmap averaging over these new probability heatmaps and obtained associated key classifier metric values. Just as before, we wrote these values to corresponding CSV files.

For median filtering, we utilized kernels of [1, 1], [3, 3] and [5, 5] across all probability heatmaps. The first of these kernels applied median filtering using solely the patch in question and therefore was equivalent to omitting median filtering overall. Likewise, we examined Gaussian

filtering standard deviations varying from zero to six in increments of one. Similar to before, we configured our filtering code to omit Gaussian filtering when provided a standard deviation value of zero. In total, we examined three options for median filtering and seven options for Gaussian filtering, or twenty-one total sets of filtering settings. We selected these options, as they allowed us to comprehend the effects of each filtering step and select optimal settings without overcomplicating the task.

Just as before, we transferred the results returned from this process to associated Excel files for further examination. We also performed several statistical tests to more rigorously examine the effects of filtering on the probability heatmaps. Namely, we carried out t-tests between the key classifier metric values without heatmap averaging and the corresponding values after heatmap averaging for each set of filtering parameters over all testing images. We then converted the resulting p-values to binary indications of statistical significance (significant or not significant) using the aforementioned level of significance and subsequently produced statistical significance heatmaps. These allowed us to examine whether the statistical significance of the effect of heatmap averaging on the key classifier metric values changed with various degrees of filtering. We prepared these statistical significance heatmaps using the R package ggplot2, such that they were easily understood and aesthetically pleasing[93].

3.4.3. Final Results

The previous stages confirmed the beneficial effects of heatmap averaging in terms of the key classifier metrics and returned optimal filtering settings for each learning rate examined. As such, we applied these steps to obtain the final probability heatmaps for each learning rate. We identified the optimal threshold in the same manner as described previously; however, we also desired to

propose a single threshold value for each key classifier metric. Therefore, we computed universal threshold values for the AUC and MAP studies by calculating the means of all corresponding optimal threshold values. We then applied both thresholds to these heatmaps to obtain the classification mappings which served as the final segmentations for this study.

We also plotted the ROC and precision-recall curves for all probability heatmaps, as well as designating the points at which the aforementioned thresholds occurred on the proper graphs. It is worth noting that we were unable to mark the position of the universal threshold on the precision-recall curve for Image 002 in the MAP study. As described in Section 3.3, we configured our code for the precision-recall curve to break upon reaching a threshold for which the model predicted no positive (tumor) patches, as this caused the denominator of the precision equation to be equal to zero[31]. Unfortunately, in the MAP study, the universal threshold value resided past this point for Image 002, thereby rendering it impossible to designate this location on the associated precision-recall curve.

Additionally, we quantified these results based on several metrics. The most notable of these was the opposite key classifier metric value for each probability heatmap. (i.e. MAP values for heatmaps optimized based on AUC and vice versa) These values provided an interesting method by which the AUC and MAP results could be compared; however, to obtain concrete results, we also performed paired t-tests for the AUC and MAP values at the corresponding learning rates.

Furthermore, we also computed metrics for the final classification mappings. Since these mappings utilized distinct measurements for these purposes (sensitivity and specificity for ROC, precision and recall for precision-recall[31]), we opted to not utilize such metrics for comparison purposes. Instead, we computed the intersection-over-union (IOU)[94] and Dice Coefficient values[95] for each classification mapping. Both of these metrics have seen extensive use for segmentation

studies in literature[11,14,16,96,97]. We also carried out paired t-tests between these values obtained with the optimal threshold and the universal threshold in both studies to examine whether one method offered statistically significantly superior results. Just as before, we wrote all comparison metric values to corresponding CSV files.

We supplemented these steps with a final set of statistical tests. These involved paired t-tests between the three final key classifier metric values and the corresponding values without filtering to examine whether the introduction of filtering in the presence of heatmap averaging had a statistically significant effect. Furthermore, we carried out additional paired t-tests between the aforementioned final key classifier metric values and their mean counterparts prior to heatmap averaging or filtering for the sake of thoroughness.

4. Results and Discussion

4.1. Model Considerations and Training Set Balancing

4.1.1. Model Performance and Architecture

Overall, the model performance was rather satisfactory. At a learning rate of 0.001, the model trained over one batch of patches in the training set in 1.8 ± 0.2* seconds, while obtaining predictions for one batch of samples in the validation set in 1.6 ± 0.2 seconds. The latter duration being shorter than its counterpart was consistent with expectations, as evaluation of patches in the validation set omitted the modification of model parameters in the backward pass. Comparably, the corresponding values for a learning rate of 0.0005 were 1.8 ± 0.1 seconds and 1.6 ± 0.1 seconds respectively. We also anticipated the slight reduction in variation from 0.001, as lower learning rates involve less aggressive variable adjustment during training and therefore more consistent training durations. Overall, this system efficiency helped us perform ten runs for each setting throughout this analysis. Regardless, if further studies demand greater computational requirements, we could improve efficiency by streamlining the code or modifying the model architecture.

*: All values reported using this format in this thesis are (mean) ± (standard deviation).

Table 2: Table of Trainable Parameters in Deep Learning Architecture

Name of Model Component	Dimensions of Model Component	Number of Trainable Parameters
Convolutional Layer 1	[3, 3, 3, 4] + [4]	112
Batch Normalization 1	[4] + [4]	8
Convolutional Layer 2	[3, 3, 4, 4] + [4]	148
Batch Normalization 2	[4] + [4]	8
Convolutional Layer 3	[3, 3, 4, 4] + [4]	148
Batch Normalization 3	[4] + [4]	8
Convolutional Layer 4	[3, 3, 4, 4] + [4]	148
Batch Normalization 4	[4] + [4]	8
Fully Connected Layer 1	[16, 100] + [100]	1,700
Fully Connected Layer 2	[100, 100] + [100]	10,100
Fully Connected Layer 3	[100, 2] + [2]	202
		12,590

Likewise, the breakdown of all trainable parameters in the adapted model architecture can be observed in Table 2 above. A few trends were immediately obvious. First, each instance of batch normalization incurred only eight trainable parameters for the model, indicating that this was a highly efficient manner of combatting overfitting. More importantly, however, was the domination of the fully connected layers in this architecture. Of the 12,590 trainable parameters in this model, 12,002 of these (i.e. ~95%) resided in the final three fully connected layers. Furthermore, the

second fully connected layer contained most of these variables, providing 10,100 (i.e. ~84%) of these parameters. Should future analyses include additional samples or techniques, this inefficient behavior could hinder performance or negatively affect the quality of the results and should therefore be corrected.

4.1.2. Training Set Balancing

The training set balancing step proved to be rather effective at achieving its goal, establishing a ratio between the number of normal patches and tumor patches in the training set of 1.0003 ± 0.0110; however, this process also incurred some serious effects on the composition of the training set. Only 25.73 ± 1.12% of all patches originally assigned to the training set remained after balancing, indicating that this process removed nearly three quarters of all original samples. This figure was even more drastic for normal patches initially placed in the training set, with merely 14.77 ± 0.75% of these persisting through balancing. This highlighted the importance of performing multiple runs for each setting to ensure that the results were not inadvertently biased. Regardless, after training set balancing, the training set only contained 69.82 ± 0.98% of all patches in the training and validation sets combined, thus reducing the training-validation split from its original value of 9:1 to just above 2:1, a far more common value in similar studies[26–28]. Finally, it is worth noting that the values presented above were obtained for a learning rate of 0.001, as these properties were measured before any model training occurred, thus rendering repetition for other learning rates redundant.

4.2. Preliminary Study

The results of the preliminary study can be observed in Figure 11 on the following page. It was immediately obvious to us that the AUC and MAP values of the model varied not only with the learning rate, but more generally with the primary image as well. As a result, it would have been rather difficult to select a learning rate which performed optimally across all primary images; however, some trends were rather clear. Most notably, the two lowest learning rates (0.0001 and 0.00005) tended to entail suboptimal results relative to their counterparts. Furthermore, the greatest learning rate consistently returned the worst predictions across the board. It is also worth noting that the error bars for this learning rate were much greater than its counterparts. Further examination revealed that this was due to the model predicting the same probability value for all patches during certain runs, resulting in AUC and MAP values of 0.5 and 0 respectively. We believed that this behavior was caused by modifying model parameters too aggressively during training due to the high learning rate.

Therefore, the most promising learning rates for both AUC and MAP were 0.001 and 0.0005. In particular, a learning rate of 0.001 appeared to offer slightly superior behavior for AUC. Conversely, a learning rate of 0.0005 presented a clear advantage for MAP, most obviously for Image 003 and Image 004. Thus, we selected these learning rates as the optimal values for the following main study.

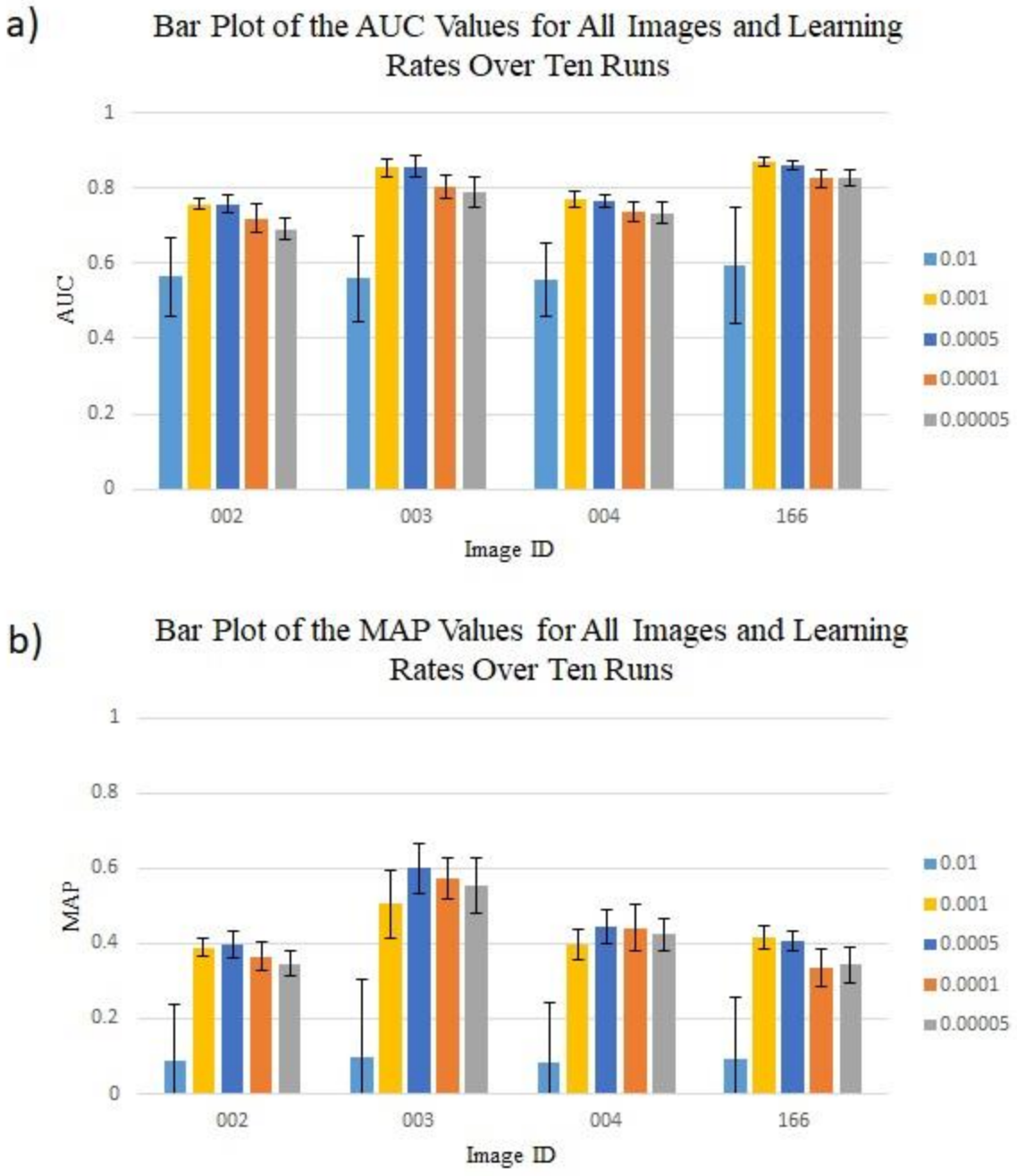


Figure 11: Bar Plots Comparing the Model Performance at Various Learning Rates Based on a) AUC and b) MAP

It is critical to note that we attempted to tune the learning rate values further as part of a coarse-to-fine approach[28,98]. More specifically, we performed additional runs for learning rates between 0.001 and 0.0005 and analyzed them in the same manner. We supplemented this with ANOVA's

for the key classifier metric values in this range on each primary image, as well as follow-up Tukey Tests wherever we identified statistical significance; however, we were generally unable to obtain statistically significant improvements within this range and the results of the aforementioned Tukey Tests were often difficult to interpret. This was likely the result of the Adam optimizer, which typically obviates the need for exhaustive hyperparameter tuning[78]. Thus, we opted to continue with the previously discussed learning rate values. We did not present these results here for the sake of brevity.

Likewise, Figure 12 presents representative probability heatmaps for each primary image at the learning rates discussed previously in this section. Just as before, some trends were immediately evident. Most notably, the patches near the edges of each heatmap tended to be darker than those closer to the center. These indicated higher tumor probabilities for interior patches, which we considered desirable behavior, as the tumors in each primary image were situated in the center. Furthermore, the model seemed to offer clearer results for Image 003 and 004 than for Image 002. The relative AUC and MAP values across all learning rates presented previously in Figure 11 corroborated this claim. Finally, patches within the gash region in Image 166 were much darker than their surroundings, indicating that the model was highly capable of identifying such patches as normal and confirming that the use of Image 166 patches for training did not hinder system performance.

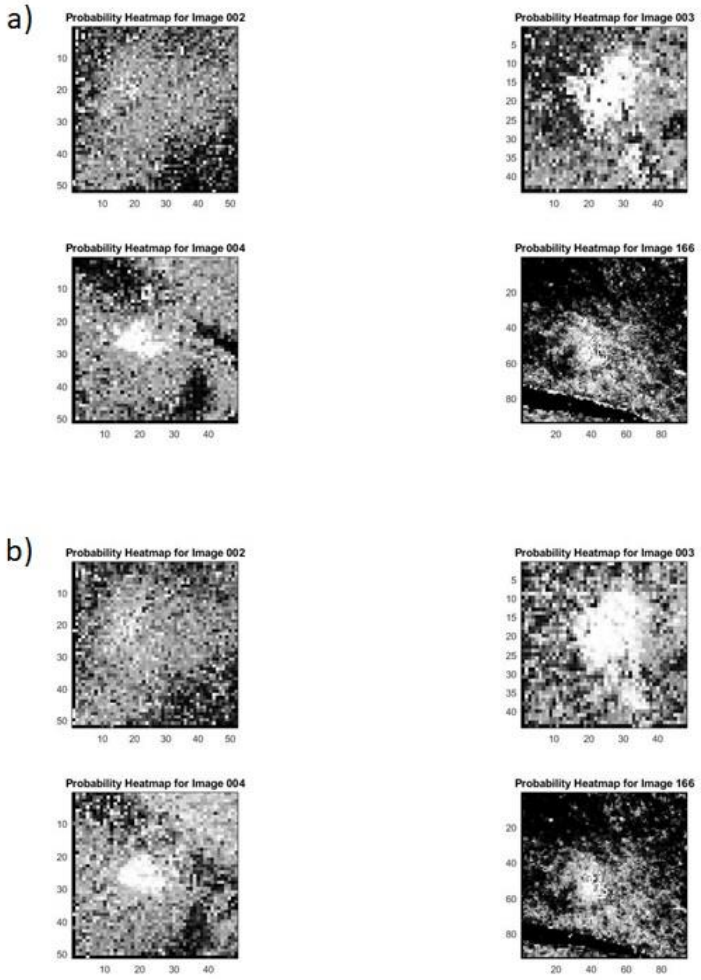


Figure 12: Representative Heatmaps for the Optimal Learning Rates for a) AUC (0.001) and b) MAP (0.0005)

4.3. Main Study

4.3.1. Heatmap Averaging

In general, heatmap averaging entailed superior results, both in terms of AUC and MAP. As shown in Figure 13 on the following page, heatmap averaging improved both metrics when Image 003 and 004 served as the testing images; however, little change occurred for Image 002 in either case. In fact, for the AUC analysis, the key classifier metric value after heatmap averaging was actually slightly lower than the mean value before heatmap averaging. It is also worth noting that

heatmap averaging produced far greater relative improvements for Image 003 and 004 in the MAP analysis than in its AUC counterpart. Regardless, just as in the preliminary study, the performance of the model varied widely with the testing image in both analyses.

The associated statistical tests performed at this stage supported these conclusions. t-tests confirmed statistically significant changes in AUC values from heatmap averaging on most test images. (p-values of 0.8133, $2.025 * 10^{-3}$ and $1.309 * 10^{-4}$ for Image 002, 003 and 004 respectively) Comparable t-tests in the MAP study produced similar results. (p-values of 0.6656, $3.028 * 10^{-5}$ and $3.697 * 10^{-5}$ on Image 002, 003 and 004 respectively) These conclusions were rather reliable, with none of the associated p-values falling near the critical value (5%); however, those for Image 003 and 004 in the MAP analysis were at least one order of magnitude lower than the corresponding values in the AUC study. This was consistent with expectations, due to the higher relative degree of improvement from heatmap averaging noted above. Finally, the paired t-tests between the key classifier metric values before and after heatmap averaging returned results that were not statistically significant, namely in the form of p-values of 0.2787 and 0.2287 for AUC and MAP respectively. The relatively minor changes observed for Image 002 were likely the cause of this insignificance.

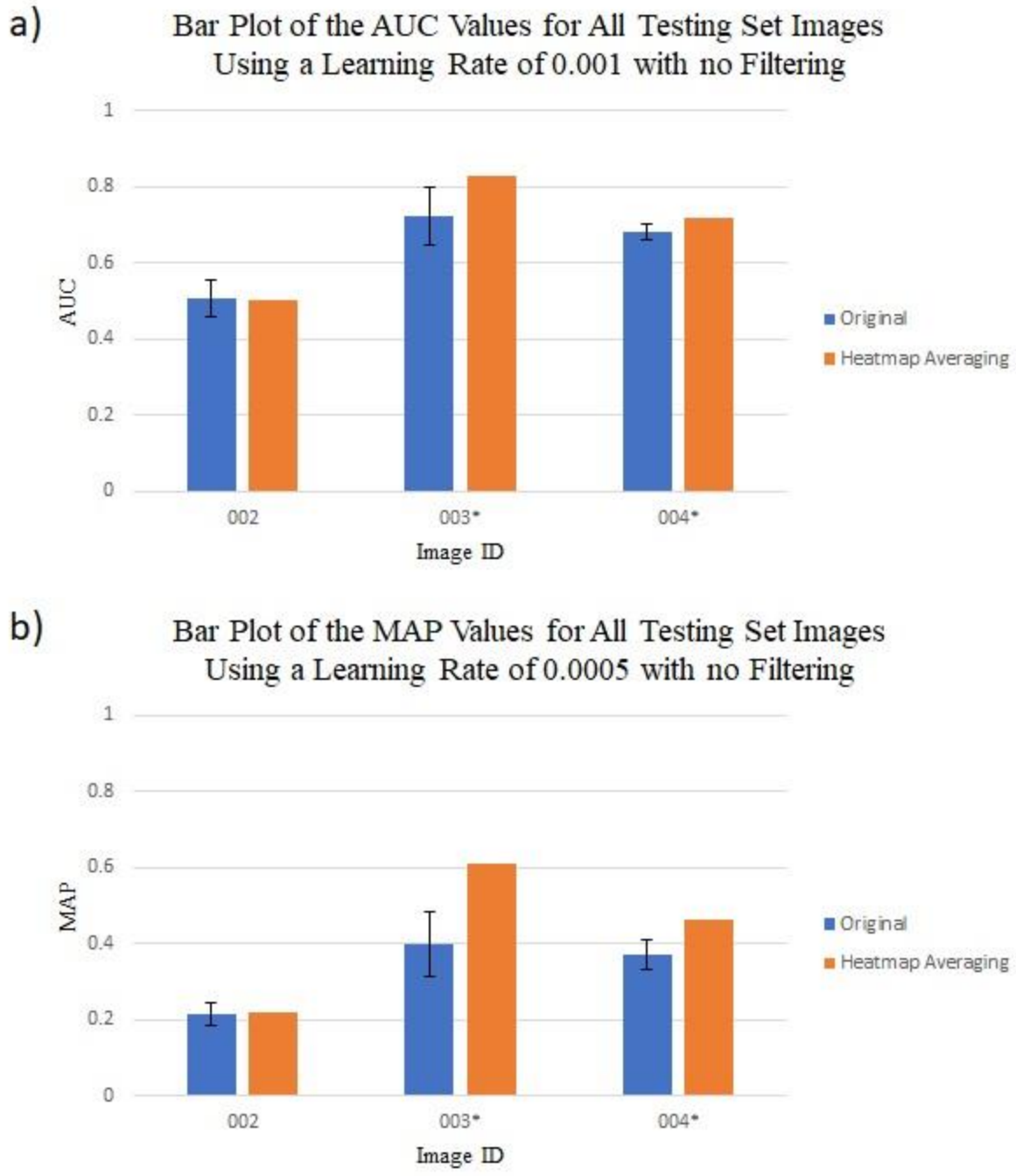


*Figure 13: Bar Plots Presenting the Effects of Heatmap Averaging on the Model Performance in Terms of a) AUC and b) MAP. Instances of statistically significant differences are indicated by *.*

Additionally, we found heatmap averaging to have a relatively minor impact on the optimal threshold values for each primary image in the AUC study, as asserted in Table 3 below. Conversely, heatmap averaging in the MAP analysis seemed to produce more drastic changes in the optimal threshold values, namely for Image 002 and 004. It is also worth noting that the optimal threshold values for Image 004 prior to heatmap averaging in the MAP study spanned a much larger range than its counterparts.

Table 3: Table of Optimal Threshold Values for Each Primary Image Before and After Heatmap Averaging

	AUC			MAP		
	Image 002	**Image 003**	**Image 004**	**Image 002**	**Image 003**	**Image 004**
No Heatmap Averaging	0.418 ± 0.057	0.847 ± 0.057	0.776 ± 0.041	0.181 ± 0.035	0.850 ± 0.081	0.650 ± 0.267
Heatmap Averaging	0.435	0.793	0.743	0.276	0.860	0.886

4.3.2. Median and Gaussian Filtering

Figure 14 presents the observed effects of median and Gaussian filtering both with and without heatmap averaging in the AUC study. In general, higher levels of filtering resulted in improved AUC values; however, the exact manner in which this occurred varied with the testing image. For Image 002, increasing degrees of Gaussian filtering with no median filtering produced a slight reduction in AUC before rapidly increasing. While this phenomenon did not occur when median filtering was introduced, the aforementioned convex trend held true. It is also worth noting that careful examination revealed that heatmap averaging consistently entailed lower AUC values than the corresponding mean points without heatmap averaging for Image 002.

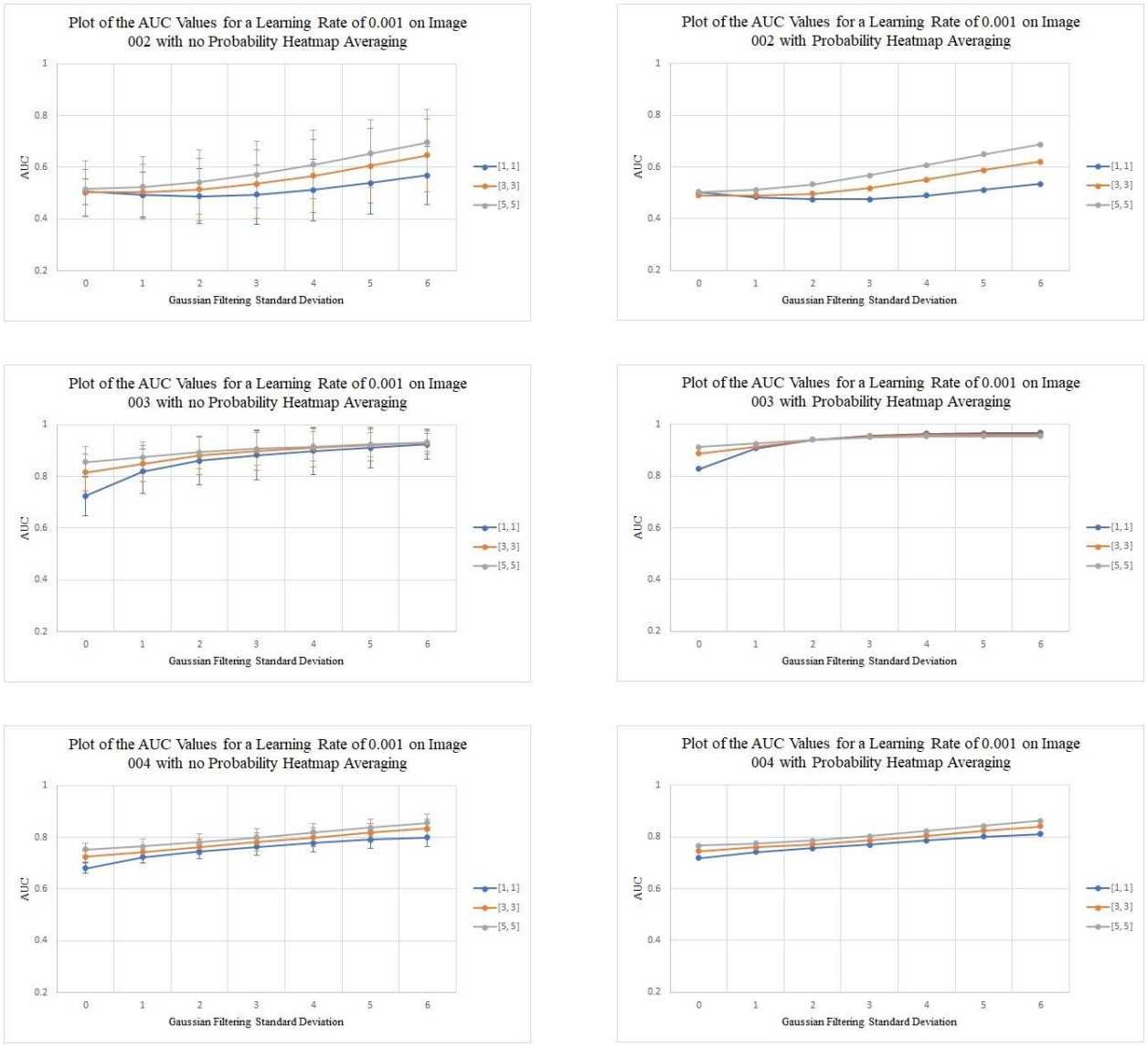


Figure 14: Plots of the AUC Values After Filtering of Various Degrees, Both With and Without Heatmap Averaging

Conversely, for Image 003, we observed diminishing returns at high filtering levels. Heatmap averaging also consistently produced higher AUC values for each combination of filtering settings over this testing image; however, this also caused some deviation from the aforementioned trend. More specifically, Gaussian filtering resulted in greater marginal increases in AUC at lower levels of median filtering. This resulted in the maximum AUC value for heatmap averaging on this image

occurring for a median filtering kernel of [1, 1] and a Gaussian filtering standard deviation of 6, which stood in contrast with the previous image. (i.e. [5, 5] and 6 respectively)

Unlike its counterparts, Image 004 consistently experienced fairly linear improvements in AUC with greater degrees of filtering. Furthermore, much like Image 003, AUC values with heatmap averaging were higher than corresponding points without heatmap averaging for every combination of filtering settings.

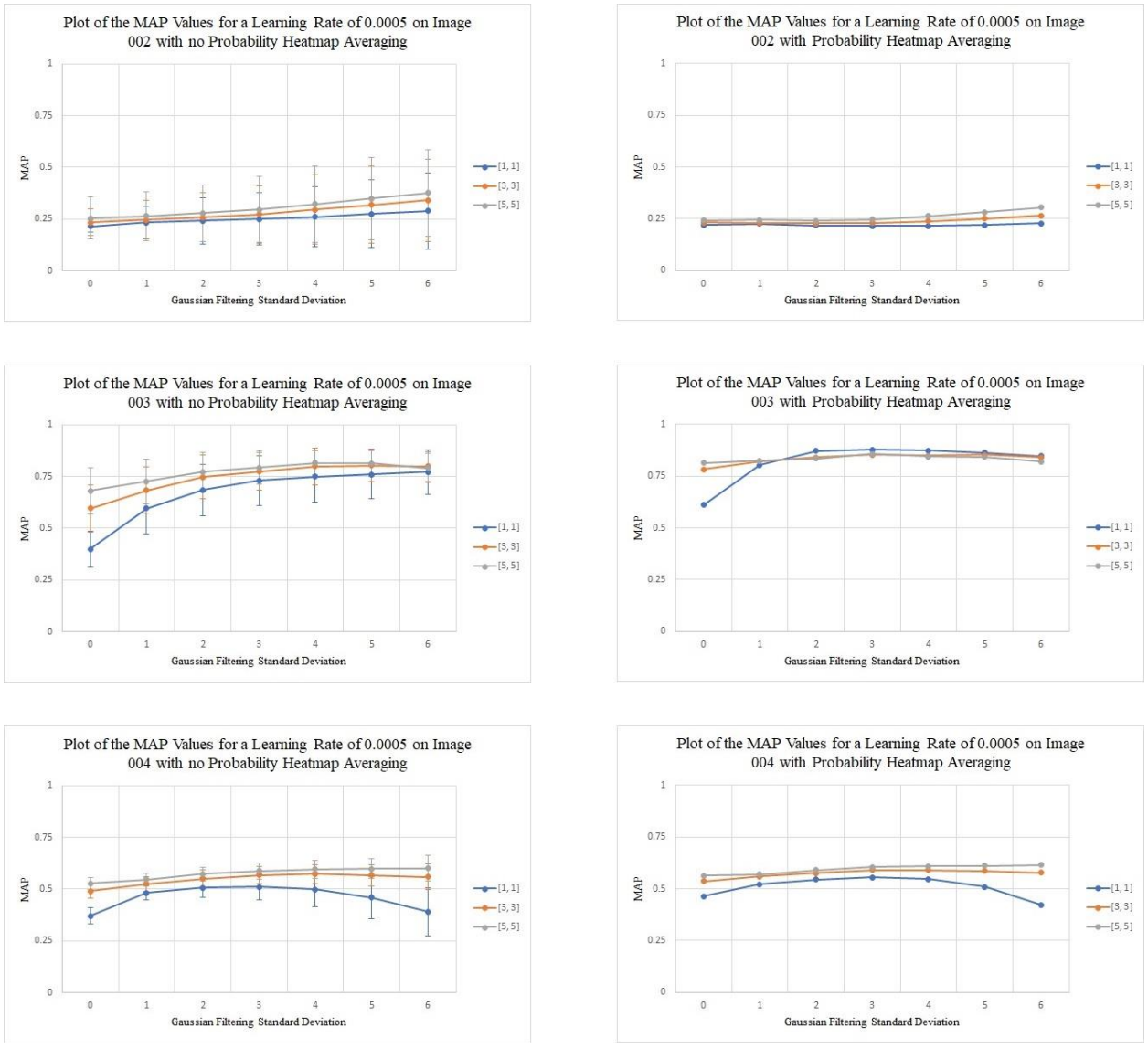


Figure 15: Plots of the MAP Values After Filtering of Various Degrees, Both with and Without Heatmap Averaging

Much like the AUC study, higher degrees of median and Gaussian filtering also tended to increase the MAP values, as asserted in Figure 15; however, the exact trends varied with the testing image in question. For Image 002, we observed the same convex growth with greater levels of filtering as in the AUC study, but it was much more gradual than before. The aforementioned dip in performance with no median filtering did not occur in this study as well. Regardless, much like before, heatmap averaging also produced lower MAP values at every combination of filtering settings besides that corresponding to no filtering. (i.e. [1, 1] and 0)

With no heatmap averaging, the MAP values for Image 003 tended to saturate at higher levels of median and Gaussian filtering in the same manner as the AUC study. Conversely, when we introduced heatmap averaging, the trend in MAP values became more far more complex. Most notably, unlike its counterparts, the maximum MAP value occurred for a median filtering kernel of [1, 1] and a Gaussian filtering standard deviation of 3. Despite this, MAP values after heatmap averaging resided above the corresponding points before heatmap averaging for all filtering settings.

The linear trend noted in the AUC study for Image 004 generally carried over to the MAP study; however, the MAP values seemed to increase more slowly with higher levels of median and Gaussian filtering. Furthermore, this did not hold true when omitting median filtering, as the MAP values increased briefly before plateauing and decreasing both with and without heatmap averaging. Regardless, just as before, heatmap averaging produced higher MAP values for every combination of filtering settings over this image.

The statistical significance heatmaps for the AUC study are presented in Figure 16. Despite AUC values before heatmap averaging residing consistently higher than those after heatmap averaging for Image 002, these changes were not statistically significant. Conversely, for Image

003, heatmap averaging produced statistically significant changes in AUC at low levels of Gaussian filtering, but these differences became insignificant as the associated standard deviation rose. Likewise, heatmap averaging entailed statistically significant changes in AUC on Image 004 only when median and Gaussian filtering were relatively minor.

In general, heatmap averaging produced (albeit mostly statistically insignificant) increases in AUC values for Image 003 and 004. Furthermore, while heatmap averaging seemed to entail a reduction in AUC values for Image 002, these changes were also statistically insignificant. Maximum AUC values occurred in all images when we applied Gaussian filtering with a standard deviation of 6. Similarly, a median filtering kernel of [5, 5] produced optimal AUC values for both Image 002 and Image 004. For these reasons, we were confident in selecting heatmap averaging, median filtering with a kernel of [5, 5] and Gaussian filtering with a standard deviation of 6 for the final parts of the AUC study.

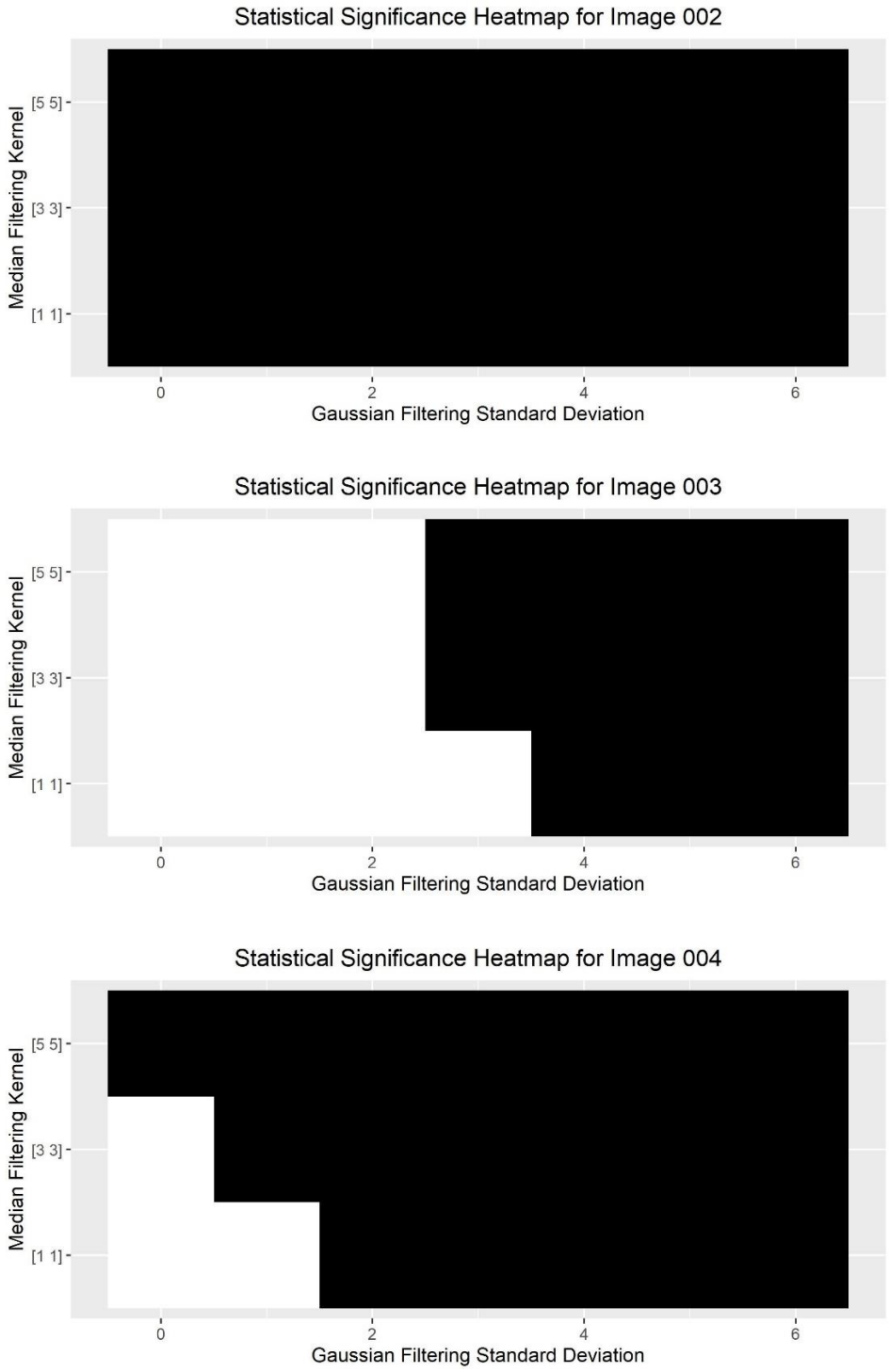


Figure 16: Statistical Significance Heatmaps for Heatmap Averaging in the AUC Study (black: not significant, white: significant)

The statistical significance heatmaps for the MAP study returned rather similar results to those of the AUC analysis, as shown in Figure 17. Just as before, heatmap averaging did not produce a statistically significant effect on MAP for Image 002 with any combination of filtering settings, despite nearly all such values being lower than their original counterparts. Conversely, for Image 003, the aforementioned trend in statistical significance became more scattered for MAP. Regardless, we did observe the general pattern of heatmap averaging only incurring statistically significant changes in MAP for lower degrees of median and Gaussian filtering. This pattern also carried over for Image 004 in the MAP study, which matched closely with the corresponding results for AUC. Before moving on, though, it is worth noting that the maximum MAP value for Image 003 occurred for a median filtering kernel of [1, 1] and a Gaussian filtering standard deviation of 3, a point at which heatmap averaging produced a statistically significant increase in MAP.

Just as in the AUC study, heatmap averaging entailed higher MAP values for Image 003 and 004, even if most such changes failed to be statistically significant. Similarly, the reductions in MAP values associated with heatmap averaging for Image 002 were also not statistically significant. Likewise, a median filtering kernel of [5, 5] and Gaussian filtering standard deviation of 6 produced optimal MAP values for Image 002 and 004. Thus, much like the AUC analysis, we also opted to utilize heatmap averaging alongside these filtering settings for the final parts of the MAP study; however, we were less confident with the decision here than for the AUC study due to the overall noisier results.

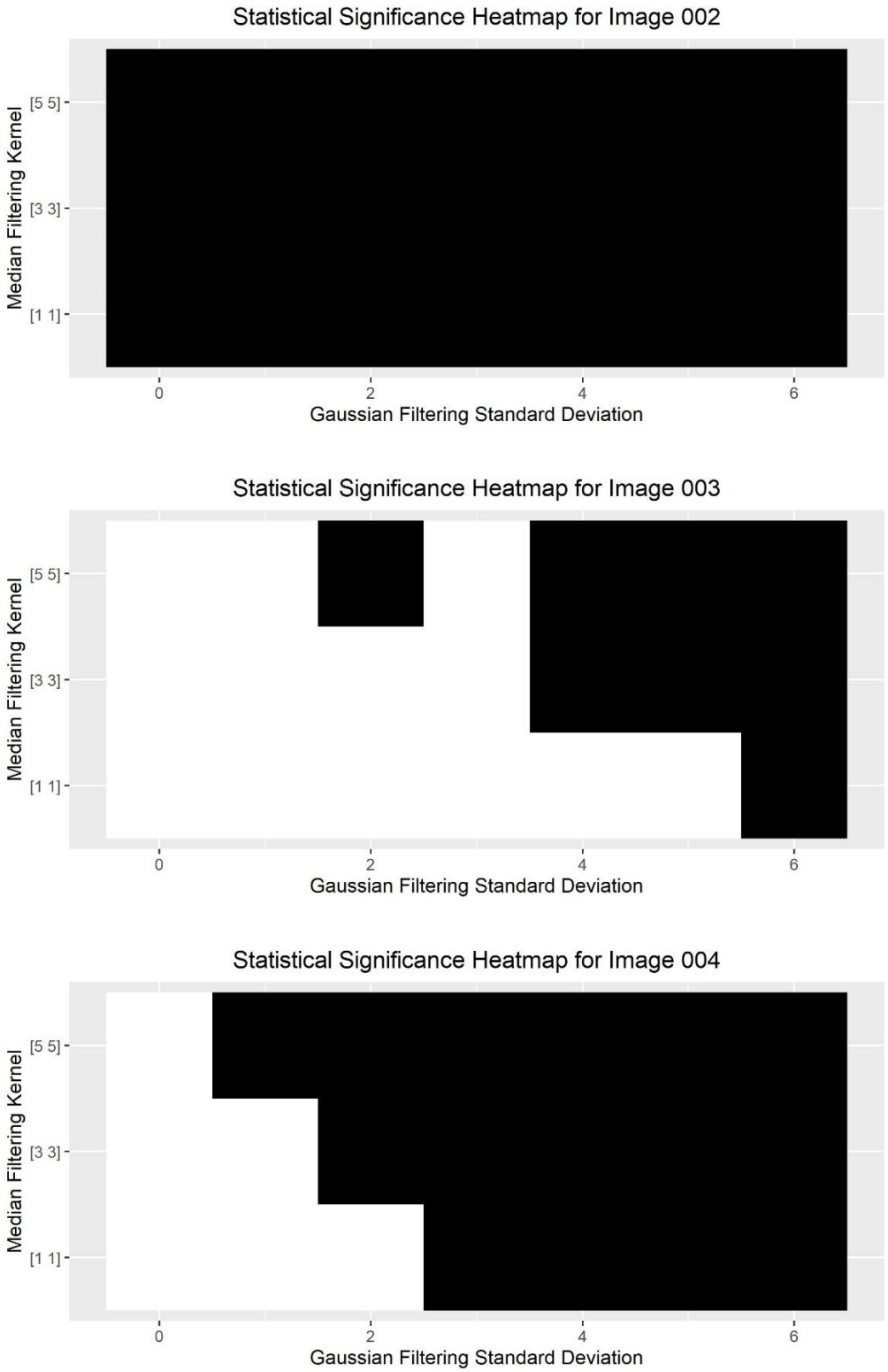


Figure 17: Statistical Significance Heatmaps for Heatmap Averaging in the MAP Study (black: not significant, white: significant)

4.3.3. Final Results

The AUC values after heatmap averaging and filtering in the AUC study were 0.6871, 0.9528 and 0.8623 for Image 002, 003 and 004 respectively. This resulted in an overall AUC of 0.8341 ± 0.1351 at the conclusion of this analysis. Thus, our pipeline struggled most with Image 002 and performed optimally on Image 003, with the results for Image 004 residing near the middle.

Figure 18 presents the final probability heatmaps and the classification mappings of the AUC study alongside the corresponding reference labels for each image. In general, patches in the interior of the classification mappings sported higher tumor probabilities, while those nearer the edges bore lower probabilities; however, the final probability heatmaps visibly varied between testing images. Applying the optimal thresholds highlighted this further. The classification mapping for Image 002 predicted a much larger tumor than the ground truth, incurring numerous false positives. Conversely, the predicted tumor boundary in Image 003 produced far fewer false positives and was overall the cleanest result in this study. Meanwhile, the final predictions for Image 004 occurred in a unique barbell shape, which warranted further examination.

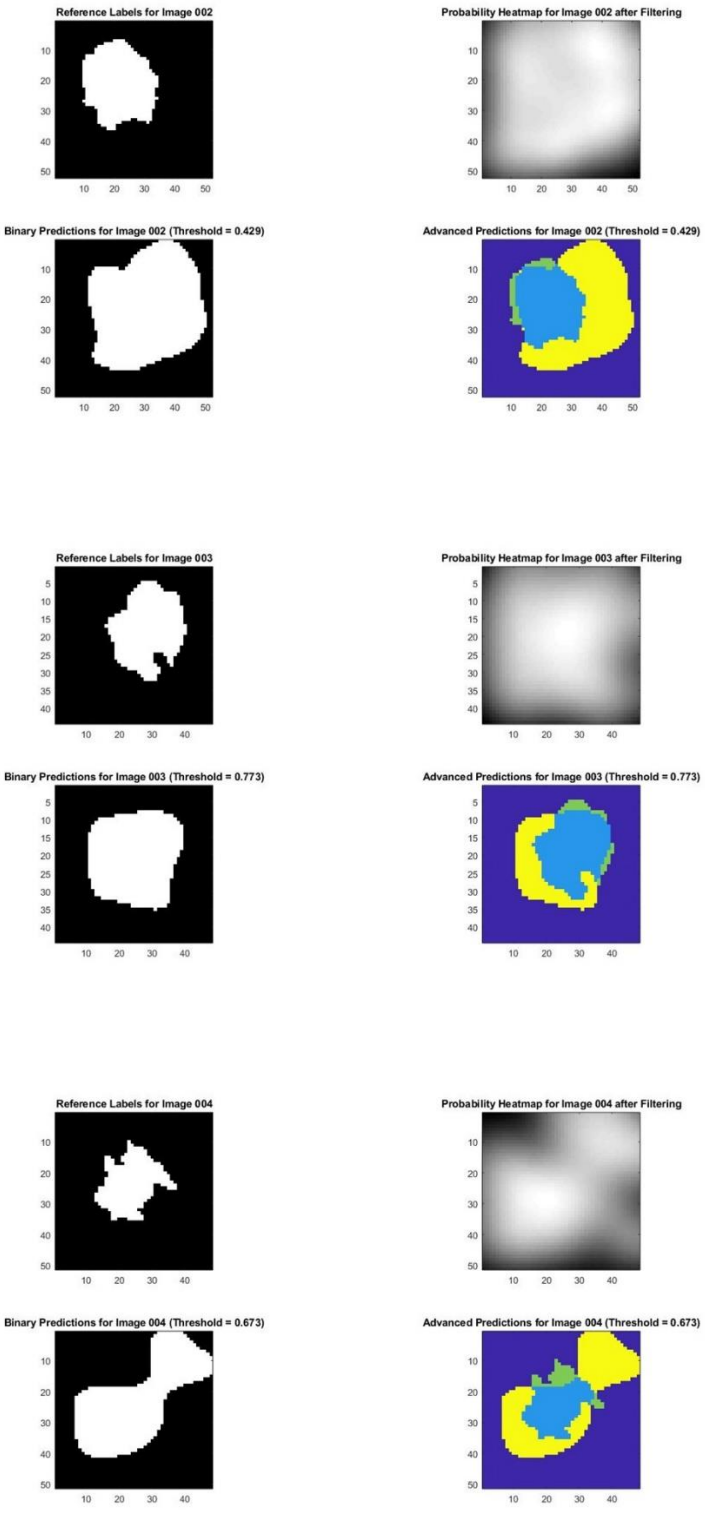


Figure 18: Reference Labels, Final Probability Heatmaps and Classification Mappings at the Optimal Thresholds for All Primary Images in the AUC Study (blue: true positive, purple: true negative, green: false negative, yellow: false positive)

The ROC and precision-recall curves for these images corroborated these results, as shown in Figure 19. The location of the optimal threshold on the ROC curve for Image 003 was far closer to ideal behavior than either of its counterparts, while the ROC curve for Image 002 took a bizarre sigmoid shape which was rather different from more typical curves[23]. The corresponding curve for Image 004 resided between these two extremes. It is also critical to note that a certain portion of the ROC curve for Image 002 resided below the straight line connecting (0, 0) and (1, 1), or the x = y line. This line denotes no discrimination in ROC space[24], indicating that, for some threshold values, the final probability heatmap for Image 002 actually behaved worse than random selections.

Comparing the locations of the universal thresholds (0.625) on these plots also provided important insight. For Image 002, this point resided in the bottom left corner. This was expected behavior relative to the optimal threshold (0.429), since the universal threshold value was greater; however, at this position, our pipeline achieved perfect accuracy on normal samples (i.e. specificity), but failed to correctly identify any tumor samples. (i.e. sensitivity)[23] Conversely, the location of the universal threshold for Image 003 was along the top edge, another logical outcome, since the corresponding value resided below its optimal counterpart (0.773). Furthermore, at this position, the model was able to correctly identify every tumor patch in the classification mapping[23]. Finally, for Image 004, the universal threshold was situated rather closely to the optimal threshold, likely due to the relative proximity of these two values (0.673 and 0.625 respectively). By extension, this position did not demonstrate the extreme behavior of either of its counterparts.

A paired t-test between the final AUC values and the corresponding mean values with no heatmap averaging or filtering returned a p-value of $6.246 * 10^{-3}$, thereby strongly confirming a statistically significant effect of heatmap averaging and filtering together on AUC. Likewise, a

similar paired t-test utilizing the AUC values with only heatmap averaging rather than the aforementioned mean values produced a p-value of 0.01326, thus also identifying a statistically significant effect of filtering on AUC when heatmap averaging was present.

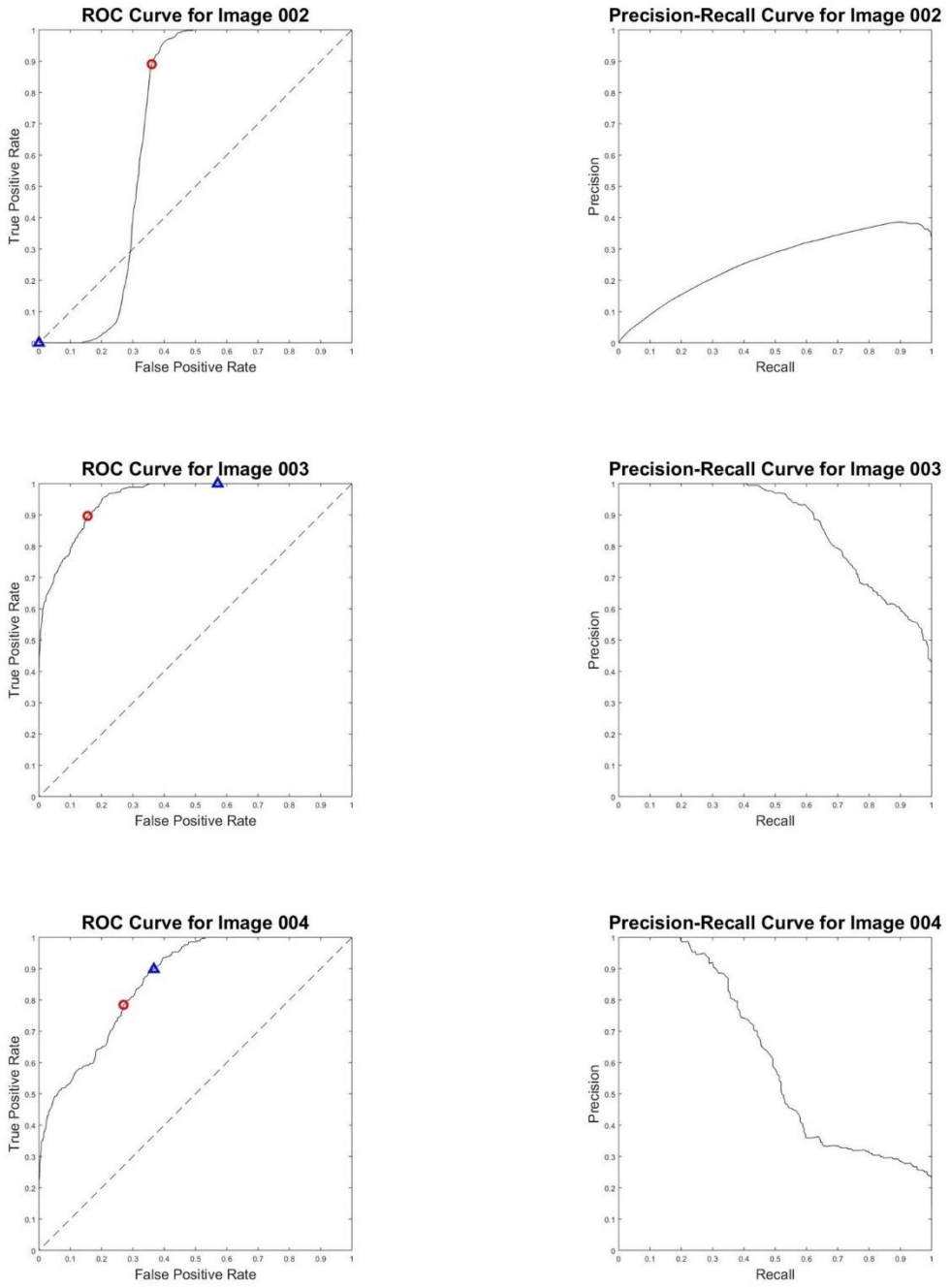


Figure 19: Final ROC and Precision-Recall Curves for All Images in the AUC Study (red circle: optimal threshold, blue triangle: universal threshold, dashed line: random behavior)

Likewise, the MAP study returned MAP values after heatmap averaging and filtering of 0.3042, 0.8204 and 0.6144 for Image 002, 003 and 004 respectively. As such, the final MAP of this study was 0.5796 ± 0.2599. These values indicated similar results as those of the AUC study. More specifically, our pipeline seemed to encounter sizeable difficulties with Image 002, perform relatively well on Image 003 and return results somewhere between these two polarities for Image 004.

The final probability heatmaps and classification mappings in the MAP study are presented alongside the associated reference labels in Figure 20. Just as before, patches near the center of the classification mappings had greater tumor probabilities, whereas our system assigned those further out lower probabilities; however, the application of the optimal thresholds caused critical differences to arise. The predicted tumor boundary for Image 002 in this study was rather similar to its counterpart in the AUC analysis, spanning a much larger portion of the classification mapping than the ground truth and thereby producing many false positives. Likewise, the classification mapping for Image 003 contained far fewer false positives than its counterparts and was cleaner than those for Image 002 or 004; however, this tumor boundary incurred more false negatives and fewer false positives than the corresponding results from the AUC analysis. As the associated probability heatmaps appeared rather similar qualitatively, this highlighted a key difference in the tumor boundaries obtained using AUC and MAP. The final predictions for Image 004 lent further support to this assertion. Despite the probability heatmap for this image in the MAP study appearing rather similar to that in the AUC analysis, applying the optimal threshold resulted in a more circular tumor boundary that incurred far more false negatives than false positives.

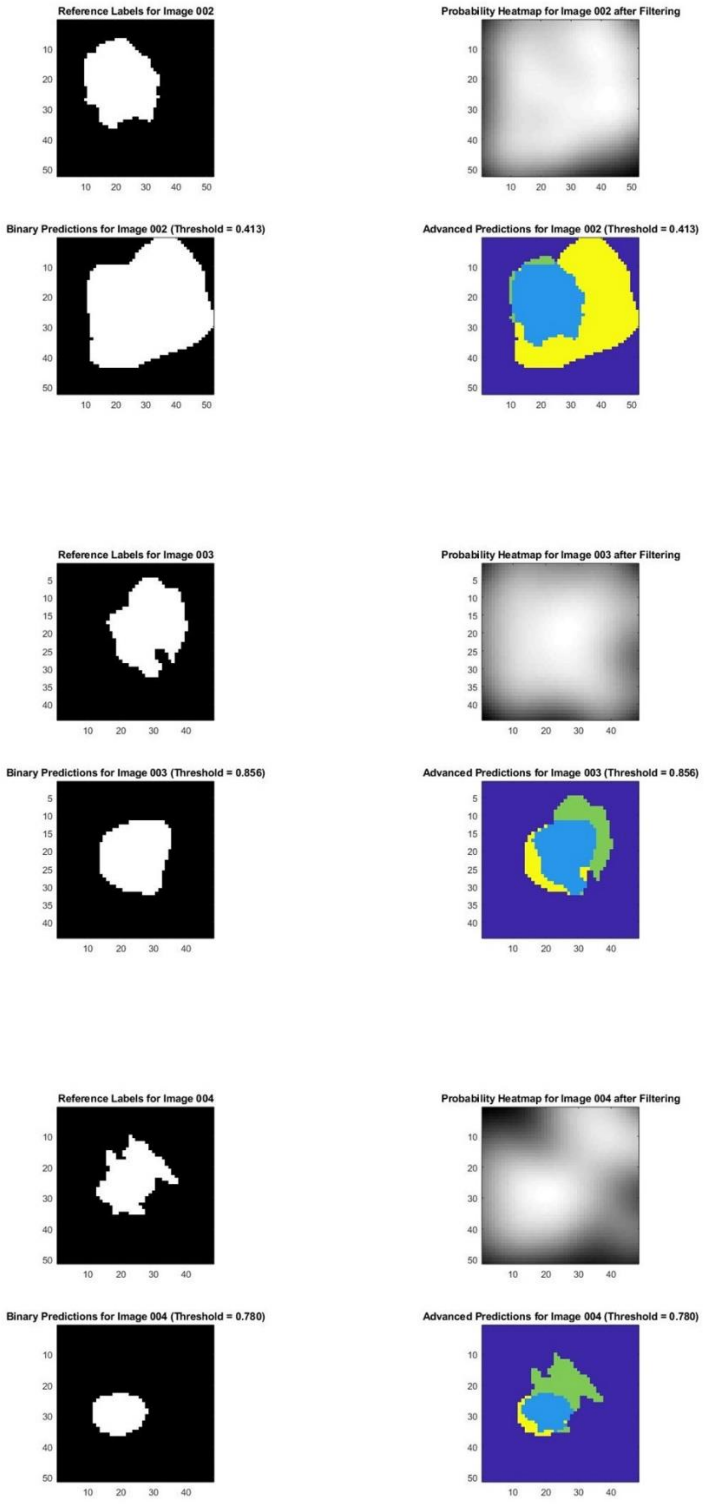


Figure 20: Reference Labels, Final Probability Heatmaps and Classification Mappings at the Optimal Thresholds for All Primary Images in the MAP Study (blue: true positive, purple: true negative, green: false negative, yellow: false positive)

Figure 21 shows the ROC and precision-recall curves for all images in the MAP study. The precision-recall curve for Image 003 drew closer to the point of ideal behavior than either of its counterparts. Conversely, the curve for Image 002 sported an unusual dome shape, which sizably differed from both the precision-recall curves of the other primary images and typical precision-recall curves in general[31]. More specifically, as the threshold increased for these predictions, the precision seemed to plateau before decreasing. Examination of the equation for precision shows that this was likely due to the number of false positives rising as the threshold increased, rather than falling as would be expected[31]. This lent further support to the conclusion that the final probability heatmap for Image 002 in this analysis offered inferior performance to random behavior at certain points. The associated curve for Image 004 lied somewhere between those of its counterparts. Finally, it is worth noting that comparing Figure 19 and Figure 21 showed that the AUC and MAP analyses produced ROC and precision-recall curves that were quite similar in shape and location. Since these studies employed identical heatmap averaging and filtering settings, this indicated that the effect of the learning rate on the final results was not as profound as we anticipated.

Much like in the AUC study, the locations of the universal threshold (0.683) on these curves suggested aggressive deviations from their optimal counterparts. As discussed previously, we were unable to designate the location of the universal threshold on the precision-recall plot for Image 002, since the associated classification mapping at this point predicted no true positives or false positives[31]; however, this extreme behavior is noteworthy regardless. Conversely, the universal threshold for Image 003 resided on the right edge. Relative to the corresponding optimal threshold point (0.856), this was expected behavior, since the universal value was sizably lower. It is also worth noting that our system achieves a recall value of one at this point, indicating that it was able

to correctly identify every available tumor patch[29]. Unlike in the AUC study, the universal threshold point for Image 004 occurred a considerable amount further down the curve from the associated optimal threshold point. This was also in line with our expectations, as the optimal threshold value here (0.780) was larger than its universal counterpart. Furthermore, the disparity between these two threshold values in the MAP study was larger than in the AUC study, thereby justifying the larger distance between them in precision-recall space. Regardless, this point did not demonstrate the polarized results observed for Image 002 and 003.

A paired t-test between the mean MAP values with no heatmap averaging or filtering and the corresponding final MAP values produced a p-value of 0.1204, thus failing to identify a statistically significant effect of heatmap averaging and filtering together on MAP. Similarly, a follow-up paired t-test in which the MAP values with only heatmap averaging replaced the mean original MAP values returned a p-value of 0.05506. This p-value resided quite closely to the selected level of significance (5%), but ultimately failed to determine a statistically significant effect of filtering on MAP when heatmap averaging was present. These results stood in contrast to the AUC study, which firmly found statistical significance in both corresponding cases. We believed that this could be due to the wider range in MAP values relative to AUC.

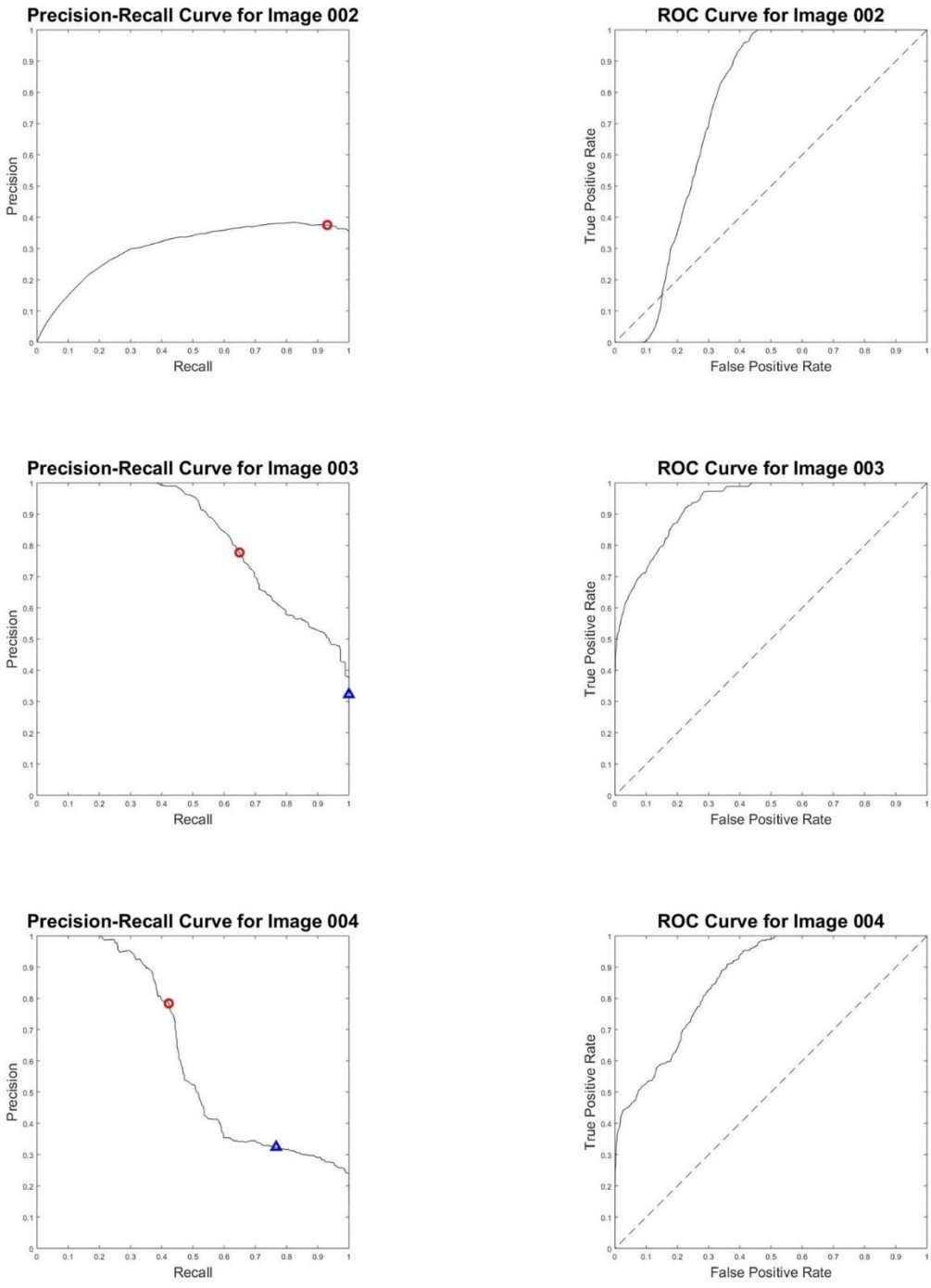


Figure 21: Final Precision-Recall and ROC Curves for All Images in the MAP Study (red circle: optimal threshold, blue triangle: universal threshold, dashed line: random behavior)

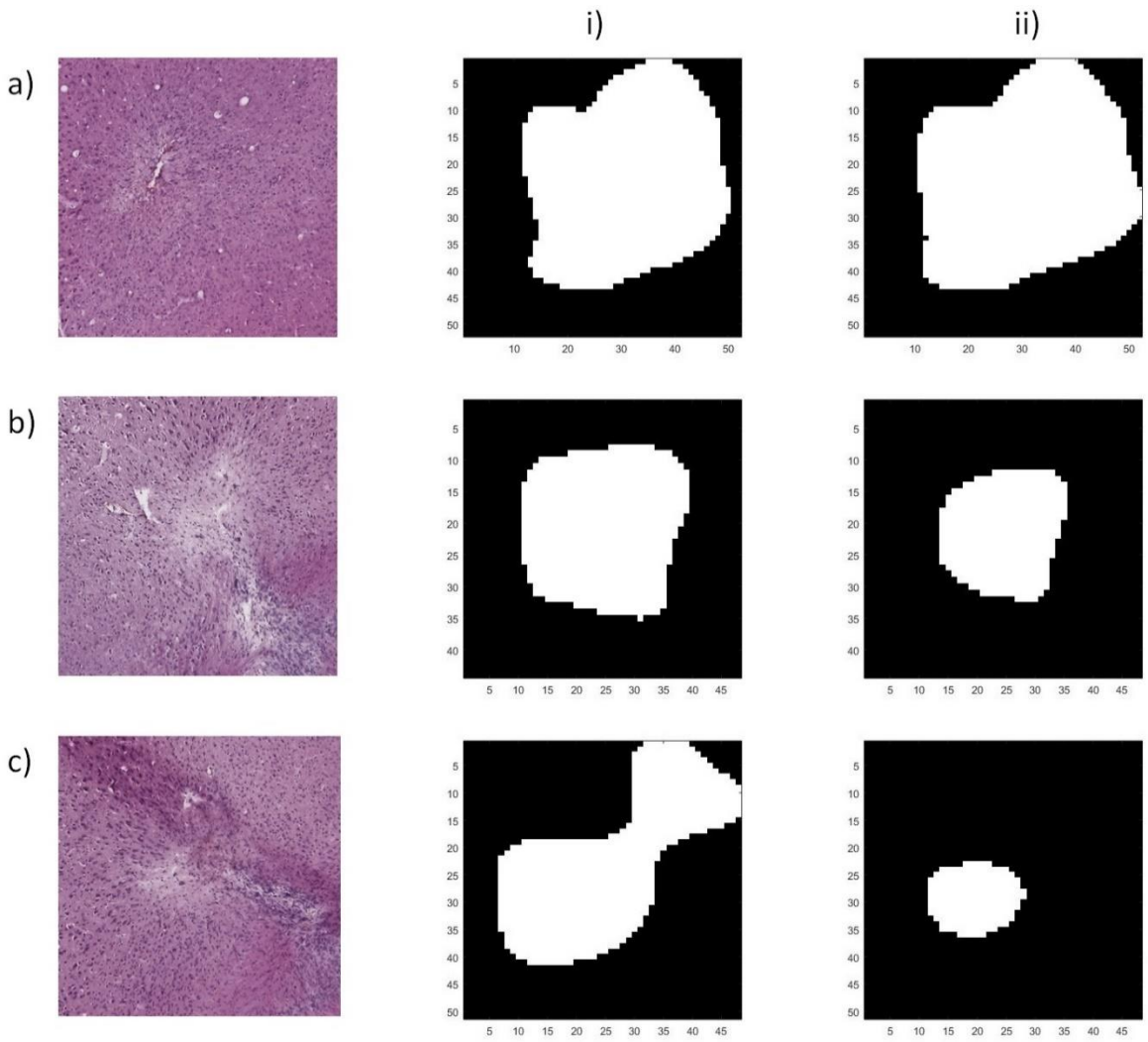


Figure 22: Final Classification Mappings Using the Optimal Thresholds for i) AUC and ii) MAP Alongside Primary Images for a) Image 002, b) 003 and c) 004

As previously stated, we desired to further examine the wide variations in system performance between the images to probe the underlying reasons. Thus, we directly juxtaposed the final classification mappings from both studies with the corresponding primary images for better comparison. This can be observed in Figure 22 above. The most obvious trend was that our pipeline was more likely to predict patches in relatively lighter regions to be tumor patches. Such a region

was most clearly present in the center of Image 003, which explained why we were able to obtain the best results on this primary image. Conversely, Image 002 did not contain this feature and was relatively consistent in tone throughout, which could justify why its key classifier metric values and classification mappings were so lacking. Finally, Image 004 contained these regions both in its center and its upper-right corner, which suggested a possible rationale for the model predicting tumor patches in these regions. Indeed, in earlier studies involving lower degrees of Gaussian filtering, we observed two separate tumor boundaries in these locations; however, upon employing higher Gaussian filtering, these two regions were joined, resulting in the barbell-shaped segmentations discussed previously.

Table 4: Summary Table of the Main Results Obtained in this Thesis. Bold p-values indicate statistical significance.

	`	AUC			MAP		
		Image 002	**Image 003**	**Image 004**	**Image 002**	**Image 003**	**Image 004**
No Heatmap Averaging	**Metric (individual)**	0.5060 ± 0.0497	0.7245 ± 0.0758	0.6807 ± 0.0189	0.2154 ± 0.0291	0.3993 ± 0.0865	0.3711 ± 0.0390
	Metric (overall)	0.6370 ± 0.1090			0.3286 ± 0.0991		
	Optimal Threshold	0.418 ± 0.057	0.847 ± 0.057	0.776 ± 0.041	0.181 ± 0.035	0.850 ± 0.081	0.650 ± 0.267
Heatmap Averaging	**Metric (individual)**	0.5021	0.8273	0.7188	0.2195	0.6096	0.4637
	Metric (overall)	0.6827 ± 0.1655			0.4309 ± 0.1971		
	p-value (individual)	0.8133	**$2.025 * 10^{-3}$**	**$1.309 * 10^{-4}$**	0.6656	**$3.028 * 10^{-5}$**	**$3.697 * 10^{-5}$**
	p-value (overall)	0.2787			0.2287		
	Optimal Threshold	0.435	0.793	0.743	0.276	0.860	0.886
Heatmap Averaging with Filtering	**Optimal Median Kernel Size (individual)**	[5 5]	[1, 1]	[5, 5]	[5, 5]	[1, 1]	[5, 5]
	Optimal Median Filtering Kernel Size (overall)	[5, 5]			[5, 5]		
	Optimal Gaussian Filtering Standard Deviation (individual)	6	6	6	6	3	6
	Optimal Gaussian Filtering Standard Deviation (overall)	6			6		
	Metric at Optimal Conditions (individual)	0.6871	0.9528	0.8623	0.3042	0.8204	0.6144
	Metric at Optimal Conditions (overall)	0.8341 ± 0.1351			0.5796 ± 0.2599		
	p-value (vs. no heatmap averaging)	**$6.246 * 10^{-3}$**			0.1204		
	p-value (vs. heatmap averaging)	**0.01326**			0.05506		
	Optimal Threshold	0.429	0.773	0.673	0.413	0.856	0.780
	Universal Threshold	0.625			0.683		

While we have established that heatmap averaging and filtering played an essential role in obtaining these results, Table 5 further supports this claim. In both the AUC and MAP studies, the original classification mappings were rather noisy, with little indication of tumor boundaries; however, these predictions for Image 002 and 003 in the AUC study tended to be more conservative with predicted tumor patches than their counterparts in the MAP analysis. Conversely, the opposite trend occurred for Image 004, where the MAP analysis produced far fewer predictions of tumor patches. Heatmap averaging reduced much of this noise for Image 003 and 004, while doing little to help for Image 002. This was consistent with our observation that heatmap averaging improved results for Image 003 and 004, while failing to do so for Image 002. Although the classification mappings for these images after heatmap averaging contained general clusters corresponding to tumors, they did not sport continuous tumor boundaries. The introduction of median and Gaussian filtering allowed us to attain these boundaries with varying degrees of accuracy in every image, highlighting the importance of heatmap averaging, median filtering and Gaussian filtering in conjunction with one another.

Table 5: Table of Figures Illustrating the Progression of Classification Mappings Utilizing the Optimal Thresholds with the Introduction of Heatmap Averaging and Filtering for Both the AUC and MAP Studies

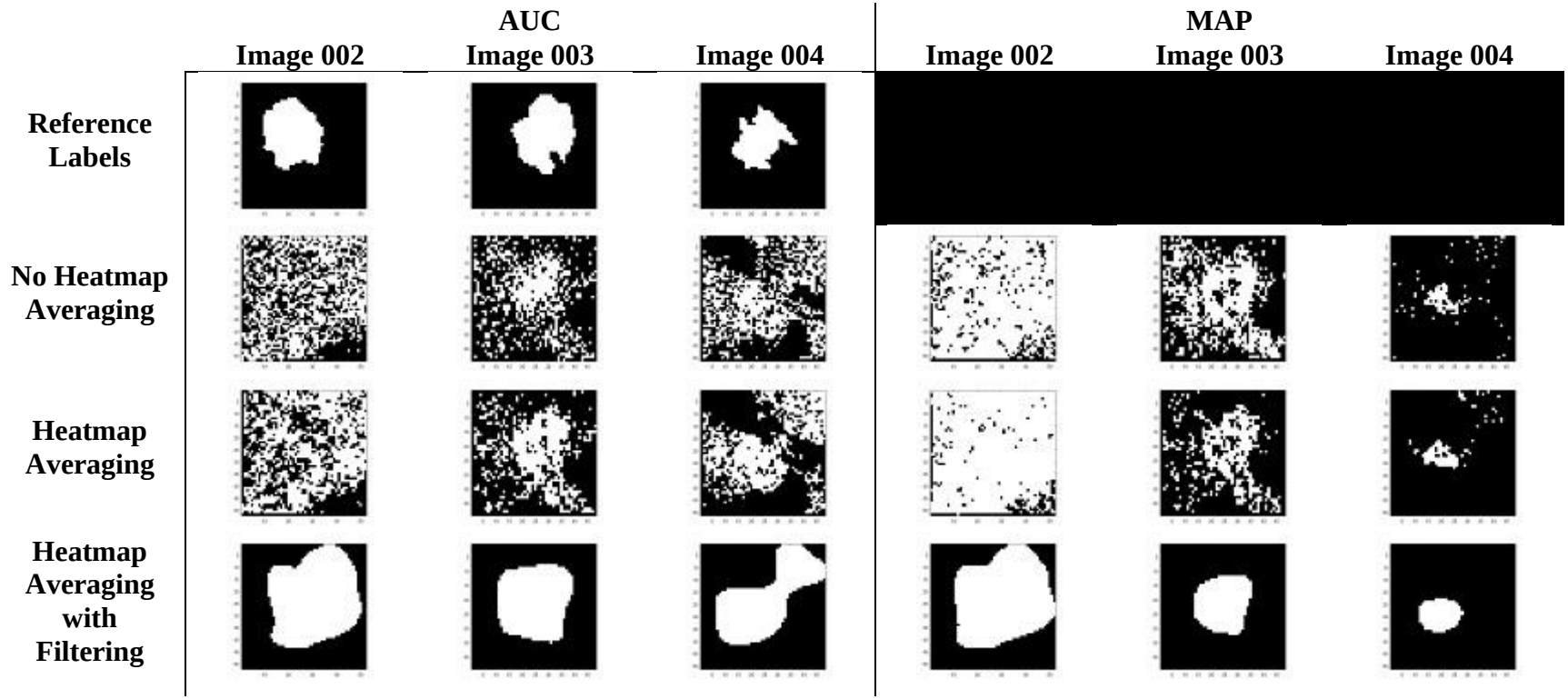


Table 6: Table of Figures Comparing the Simple Binary Classification Mappings for Both Studies Using the Optimal Thresholds and the Universal Thresholds

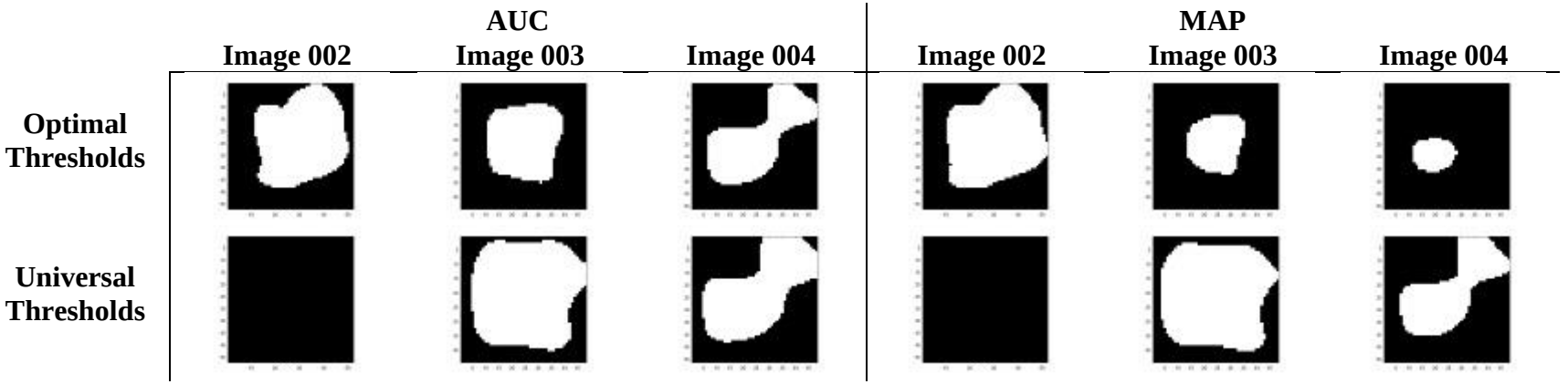
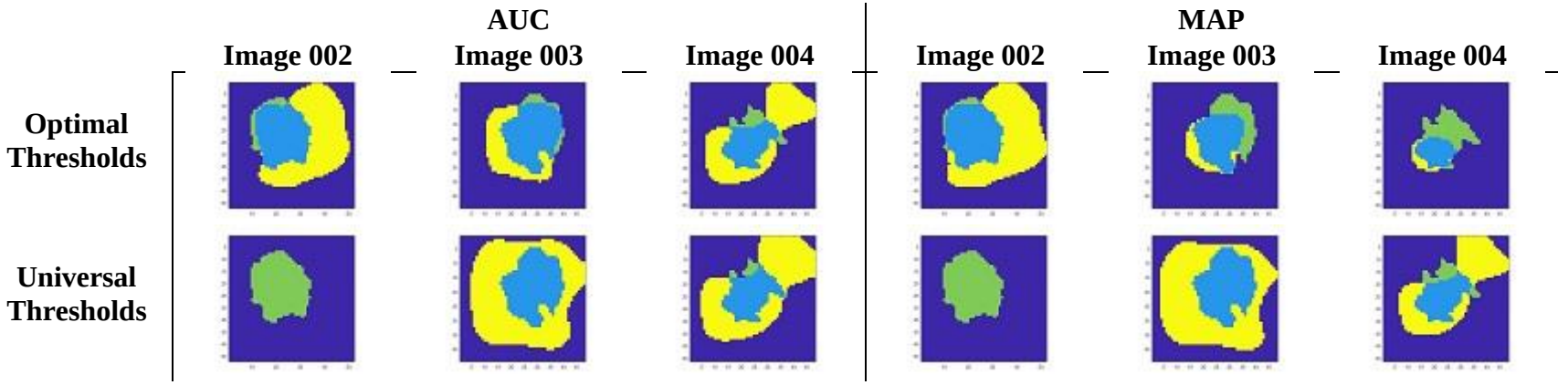


Table 7: Table of Figures Comparing the Advanced Color-Coded Classification Mappings for Both Studies Using the Optimal Thresholds and the Universal Thresholds

(blue: true positive, purple: true negative, green: false negative, yellow: false positive)



The effects of applying the universal thresholds over their optimal counterparts can be observed in Table 6 and Table 7 on the previous page. In general, employing these universal values severely impacted the qualities of the resulting segmentations. This was most evident for Image 002, as the higher thresholds caused the models to predict solely normal patches in both cases. This allowed them to achieve perfect specificity, as discussed previously; however, the practical considerations of predicting the complete lack of a tumor in an image containing one are rather serious, especially in a biomedical setting with relatively low tolerance for error. Conversely, the predicted tumor boundaries for Image 003 using the universal thresholds were spread over far larger areas than their counterparts with the optimal threshold, thus incurring a large number of false positives. This allowed the models to correctly identify all available tumor patches. By comparison, the novel classification mappings for Image 004 bore the same unique shape as the results using the optimal threshold in the AUC study; however, this was rather distinct from the much smaller original tumor predicted in the MAP study. This disparity in behavior and shape between the AUC and MAP studies was unique to this image, as was its lack of extreme behavior. It is also interesting to note that the results obtained on Image 004 using the optimal threshold in the AUC study and the universal threshold in the MAP study were remarkably similar. As these threshold values differed by only 0.010 (0.673 and 0.683 respectively), this offered further support to the conclusion that our system was not particularly sensitive to changes in the learning rate. In general, however, standardizing the threshold value produced aggressive and largely negative changes in the majority of classification mappings in both studies, indicating that utilizing a single threshold for future studies would produce subpar results.

Both key classifier metric values in the AUC and MAP studies are presented together in Table 8 on the following page. In general, these studies produced rather similar metrics for the overall classifiers. The higher learning rate for the AUC study returned visibly lower AUC and MAP values on Image 002, but both of these values for Image 003 were higher than their MAP counterparts. Meanwhile, the AUC and MAP values for Image 004 in both studies were identical to two decimal places, suggesting that these results were not particularly sensitive to changes in the learning rate. These results further corroborated the conclusion that improvements in the AUC and MAP values from fine-tuning the learning rate were minor; however, associated t-tests for these properties all but confirmed this assertion. Namely, paired t-tests between the AUC and MAP values at these learning rates produced p-values of 0.6005 and 0.9945 respectively, both of which were very distant from the selected level of significance. (5%) As discussed in Section 4.2, this was consistent with the Adam optimizer's documented ability to mitigate the need for hyperparameter tuning[78].

Table 8: Table of AUC and MAP Values in Both Studies. Key classifier metrics examined in each study are presented in bold.

	AUC			**MAP**		
	Image 002	**Image 003**	**Image 004**	**Image 002**	**Image 003**	**Image 004**
AUC	**0.6871**	**0.9528**	**0.8623**	0.7503	0.9342	0.8633
MAP	0.2605	0.8648	0.6131	**0.3042**	**0.8204**	**0.6144**

Conversely, the IOU and Dice Coefficient values obtained in both studies utilizing the two distinct threshold values can be observed in Table 9 and Table 10 respectively. The most immediate observation was the classification mappings for Image 002 which utilized the universal threshold attaining values of zero for both metrics. Careful examination of the equations for these properties revealed that this was due to the complete lack of true positives, or correctly identified tumor patches[16,99]. In general, applying the universal threshold values produced lower IOU and Dice Coefficient values than their optimal counterparts for all primary images in both studies. This disparity was lowest for Image 004 in the AUC study, likely due to the aforementioned proximity of its optimal threshold value to the corresponding universal value. Meanwhile, when the optimal threshold was applied, these values for Image 003 were higher in the AUC study, while the MAP study offered superior results for Image 004. Comparably, these metric values for Image 002 in the AUC analysis resided within 0.01 of their counterparts for MAP, which we believed was likely the result of the overall similarities in shape and size between the corresponding classification mappings. Finally, it is worth noting that the IOU and Dice Coefficient values for Image 004 with the optimal threshold in the AUC study were identical to those with the universal threshold in the MAP study to three decimal places, further highlighting the similarities between these two results.

Table 9: Table of IOU Values for the Classification Mappings Obtained in Both Studies Using the Optimal Thresholds and the Universal Thresholds

	AUC			**MAP**		
	Image 002	**Image 003**	**Image 004**	**Image 002**	**Image 003**	**Image 004**
Optimal	0.3683	0.5668	0.2950	0.3651	0.5474	0.3786
Universal	0	0.3185	0.2760	0	0.3227	0.2948

Table 10: Table of Dice Coefficient Values for the Classification Mappings Obtained in Both Studies Using the Optimal Thresholds and the Universal Thresholds

	AUC			MAP		
	Image 002	**Image 003**	**Image 004**	**Image 002**	**Image 003**	**Image 004**
Optimal	0.5383	0.7235	0.4556	0.5349	0.7075	0.5492
Universal	0	0.4832	0.4326	0	0.4879	0.4554

Paired t-tests comparing metric values at the optimal thresholds and universal threshold in the AUC study produced p-values of 0.1747 and 0.2155 for IOU and Dice Coefficient respectively, thereby firmly failing to determine statistically significant differences. Likewise, comparable t-tests for the corresponding values in the MAP study resulted in p-values of 0.1097 and 0.1639 for IOU and Dice Coefficient respectively. The aforementioned lack of statistical significance was likely the product of the relatively minor changes in these values for Image 004. Regardless, the use of a single threshold across all images generally produced aggressive and largely negative effects in the resulting segmentations qualitatively.

Overall, both key classifier metrics offered satisfactory results. For reference, *Xu et al.* examined various approaches incorporating CNN's for segmentation of stained breast and colorectal cancer images, which returned AUC values between 0.83115 and 0.93163[28]. Likewise, *Litjens et al.* reported AUC values spanning from 0.74 to 0.90 in their study classifying and segmenting stained sentinel lymph node images[27]. Therefore, our AUC's of 0.8341 ± 0.1351 resided comfortably within the expected range of values, albeit on the lower end. Conversely, as previously stated, MAP has yet to gain traction in biomedical image segmentation studies; however, *Mercan et al.* attained MAP values of 0.5727, 0.6755 and 0.7136 when segmenting

tumors in breast cancer histology images using CNN's[58]. By comparison, our final MAP values of 0.5796 ± 0.2599 lied just above the lower bound of this range.

Therefore, we are confident in recommending the use of both AUC and MAP for future studies; however, in general, AUC seemed to offer slightly more desirable behavior. We identified statistically significant effects on AUC from both heatmap averaging and filtering together and filtering in the presence of heatmap averaging through two corresponding paired t-tests. As shown in Table 5, this was consistent with expectations, since both steps were imperative for removing noise and delineating the tumor boundaries. Conversely, the corresponding paired t-tests for the MAP study failed to identify statistically significant effects. Similarly, as discussed in the previous section on filtering, the statistical significance heatmaps for heatmap averaging at various degrees of filtering were more scattered in the MAP study and the corresponding results were more difficult to interpret. Finally, the optimal thresholds for each primary image fluctuated far more with the addition of heatmap averaging and filtering in the MAP study. Regardless, the use of MAP over AUC resulted in a lower number of false positives, which could allocate a niche for this metric moving forward. For example, if this system was employed alongside a tumor resection operation, the reduced frequency of false positives using MAP could minimize the amount of healthy tissue erroneously classified as cancerous and subsequently removed.

5. Conclusions and Future Work

In conclusion, our pipeline was shown to effectively delineate tumor boundaries in histopathology images of glioblastoma; however, the performance varied widely throughout and seemed to depend on the color distributions of the test images. The introduction of heatmap averaging efficiently removed noise from the primary probability heatmaps and statistically significantly improved the AUC and MAP values for the majority of images analyzed. Likewise, increasing amounts of median and Gaussian filtering tended to further improve the AUC and MAP values. The selected levels of filtering in conjunction with heatmap averaging allowed us to obtain continuous tumor boundaries upon applying the optimal thresholds. While both studies were robust to the use of a universal threshold value, the resulting segmentations were severely degraded in accuracy. This degraded performance with a universal threshold was the major limitation of the present thesis and requires further investigation. Optimizing this system based on AUC and MAP returned very similar results; however, MAP produced tumor boundaries which incurred fewer false positives at the cost of more false negatives. MAP also returned less desirable results than AUC throughout. Therefore, we suggest the use of AUC moving forward, while reserving MAP for cases where it is imperative to control the number or frequency of false positives.

The most obvious direction for future work is the use of additional histology images. Similar studies in literature involved far more primary images than those examined here[14,27,49,28,100]. The most prominent source of such histology images is The Cancer Genome Atlas (TCGA)[101], which has seen use in several related studies[14,17,25,73]. The incorporation of such images offers two possible directions. First, we could employ more objective expert labeling for both the existing images and additional ones. This would obviate interobserver variability[102] and therefore offer a

clearer idea of how our system scales up. Alternatively, we could obtain reference labels from other histologists for this data. Despite attempts to standardize tumor grading and labeling[102,103], interobserver variability remains a persistent issue in histopathological analysis[104,105]. Evaluating and optimizing our pipeline using ground truths obtained from multiple sources would allow us to examine this phenomenon from a unique perspective.

Including a larger number of primary images would also allow us to designate one or more entire images as the validation set, which would enable an overhaul of our experimental design. As is, our system used the results on the testing set to evaluate whether to include heatmap averaging and to what degree median and Gaussian filtering should be applied. One could argue that we should have assigned these tasks to the validation set and simply applied the optimal settings to the testing set at the end. Additionally, as discussed previously, we only used the results on the validation set to roughly tune the model hyperparameters. Thus, designating the aforementioned tasks to the validation set would incur a marginal downside in exchange for a more sizable benefit. The main obstacle preventing us from doing so here was the composition of the validation set, which we synthesized using random patches from three primary images. Furthermore, we had produced most such patches using data augmentation. Thus, we could not examine the effects of median or Gaussian filtering using this set, since these steps relied on the probabilities of neighboring patches; however, if one designated original patches from entire primary images as the validation set, this restriction would no longer apply. For these reasons, we propose that future analyses which make use of additional histology images designate primary patches from some of these images as the validation set, utilize this novel set for tuning the learning rate, evaluating heatmap averaging and selecting optimal filtering parameters and subsequently apply the resulting properties over the testing set.

Unfortunately, our attempts to establish universal thresholds produced subpar results in both the AUC and MAP studies. Due to the sizeable variation in the locations of the optimal thresholds between testing images, applying the universal thresholds generated negative effects on the resulting segmentations. A possible alternative method would be to simply utilize a universal threshold of 0.5, as we employed on the validation set, and iteratively tune this parameter based on the observed tumor boundaries.

We could also utilize distinct criteria for determining the optimal thresholds. One possible option would be to set a hard cap on certain properties. For example, when examining AUC, one could specify that the false positive rate must not exceed 5%, as in statistical hypothesis testing[81], and utilize the minimum threshold which achieved this condition; however, the most obvious alternative for AUC studies is the Youden index, which has seen widespread use in biomedical diagnostic studies[106]. First proposed in 1950[107], this metric represents the point on the ROC curve that maximizes the distance from the x = y line[108], which denotes random behavior[24]. Furthermore, this quantity has seen successful use in biomedical imaging studies employing CNN's[109,110]. An example of the location of the Youden index in ROC space is presented in Figure 23 on the following page[108].

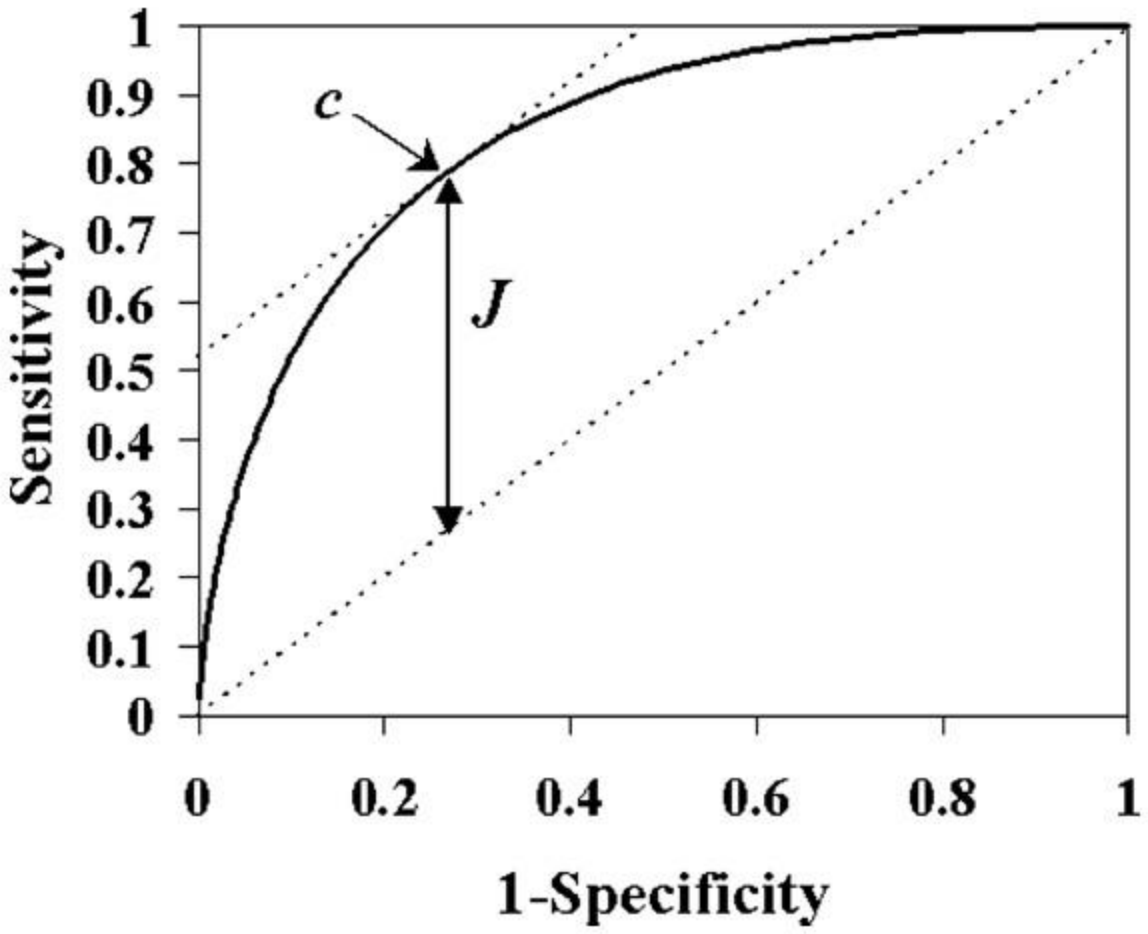


Figure 23: Diagram Indicating the Youden Index (J) for a Sample ROC Curve[108]

As previously stated, the performance of our system seemed to rely heavily on the color distributions of the testing images. Therefore, future studies should aim to remedy this via data augmentation. In addition to the rotations and flipping applied in this analysis, further augmentation can involve color variations[47,50,57,73]. To improve effectiveness, this step can occur after converting the RGB images into the H&E color domain[111]. Furthermore, modifying our method of data augmentation to combine rotation and flipping in series would allow one to increase the number of available patches further[27,57]. For example, applying three 90° rotations after vertical

flipping or horizontal flipping would increase the number of patches by a factor of twelve, rather than the six-fold increase presented in this thesis.

Furthermore, we could attempt to improve the results via transfer learning[112]. In 2014, *Yosinki et al.* confirmed the benefits of pretraining neural nets on large available datasets and fine-tuning the parameters using the data in question[113]. Additional research has proven this practice to be rather beneficial in biological image analyses similar to this one as well[53,63,114]. We propose using the CIFAR-10 and CIFAR-100 datasets of small color images for these purposes[115]. This is both due to their ubiquity as benchmarks for deep learning studies[116] and their similar image size to those used in this analysis (i.e. 32 x 32[115]); however, since our model was inflexible to changes in the input size, one would need to repeat the preprocessing step with this new window size and adjust the model architecture accordingly. Some sample images in the former dataset are presented in Figure 24 below[71].

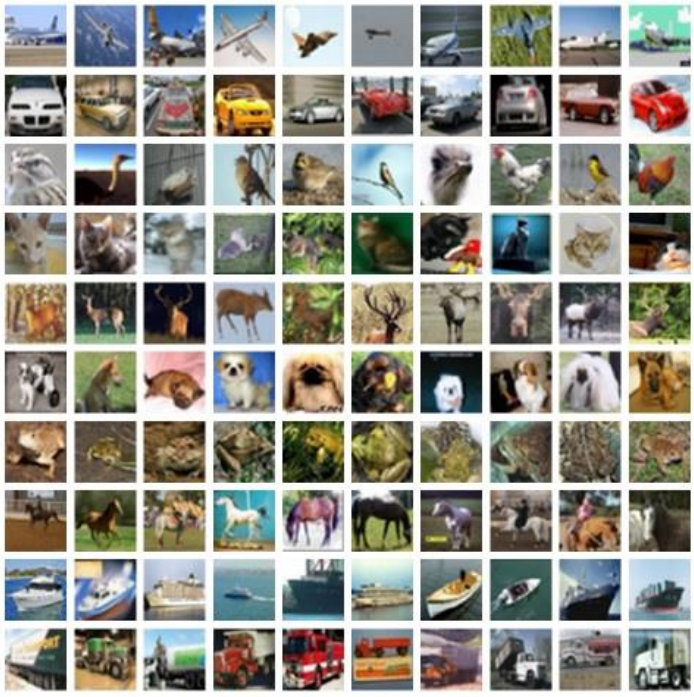


Figure 24: Sample Images from All Classes in the CIFAR-10 Dataset[71]

Additionally, one could evaluate the performance of a distinct model architecture. The most promising of these would likely be the Inception architecture, which was first developed in 2014 to rival AlexNet[117]. Since then, it has seen use across various studies similar to this one, often directly in comparison to AlexNet[26,51,57,63,64]. ResNets are another potential option, as they are effective at combatting the pervasive vanishing gradient problem in deep learning[118]. These have also produced very strong results in histopathological imaging studies[21,25]; however, it is critical to note that the vanishing gradient problem is most obstructive for very deep neural networks[119]. Since the model analyzed in this thesis was relatively shallow, the utility of ResNets is questionable here.

Finally, while we have clearly shown the beneficial effects of simple heatmap averaging, we could further refine this step moving forward. Many studies in literature, both early stage and modern, suggested or demonstrated improvements from utilizing a weighted averaging scheme over an unweighted one[25,83,84,120]. Indeed, earlier in this project, we observed additional improvements using such a method, where we assigned greater weights to heatmaps which sported higher key classifier metric values; however, after more careful deliberation, we found these results to be artificially inflating our system performance, since these measures would not be possible in the absence of a labeled ground truth. Therefore, we removed this step for our core analysis. We elaborate on this in Appendix 7.2. A viable alternative would be computing the weights using the final training set errors[83], rather than the key classifier metric values.

6. References

1. Fitzmaurice C, Allen C, Barber RM, et al. Global, Regional, and National Cancer Incidence, Mortality, Years of Life Lost, Years Lived With Disability, and Disability-Adjusted Life-years for 32 Cancer Groups, 1990 to 2015. *JAMA Oncol*. 2017;3(4):524. doi:10.1001/jamaoncol.2016.5688.
2. Patel AP, Fisher JL, Nichols E, et al. Global, regional, and national burden of brain and other CNS cancer, 1990–2016: a systematic analysis for the Global Burden of Disease Study 2016. *Lancet Neurol*. 2019;18(4):376-393. doi:10.1016/S1474-4422(18)30468-X.
3. Leece R, Xu J, Ostrom QT, Chen Y, Kruchko C, Barnholtz-Sloan JS. Global incidence of malignant brain and other central nervous system tumors by histology, 2003-2007. *Neuro Oncol*. 2017;19(11):1553-1564. doi:10.1093/neuonc/nox091.
4. Shen D, Wu G, Suk H-I. Deep Learning in Medical Image Analysis. *Annu Rev Biomed Eng*. 2017;19(1):221-248. doi:10.1146/annurev-bioeng-071516-044442.
5. Litjens G, Kooi T, Bejnordi BE, et al. A survey on deep learning in medical image analysis. *Med Image Anal*. 2017;42:60-88. doi:10.1016/j.media.2017.07.005.
6. Komura D, Ishikawa S. Machine Learning Methods for Histopathological Image Analysis. *Comput Struct Biotechnol J*. 2018;16:34-42. doi:10.1016/j.csbj.2018.01.001.
7. LeCun Y, Bottou L, Bengio Y, Haffner P. Gradient-Based Learning Applied to Document Recognition. *Proc IEEE*. 1998;86(11):2278-2323. doi:10.1109/5.726791.
8. Xu Y, Jia Z, Wang LB, et al. Large scale tissue histopathology image classification, segmentation, and visualization via deep convolutional activation features. *BMC Bioinformatics*. 2017;18(1). doi:10.1186/s12859-017-1685-x.

9. Bhandarkar SM, Koh J, Suk M. Multiscale image segmentation using a hierarchical self-organizing map. *Neurocomputing*. 1997;14(3):241-272. doi:10.1016/S0925-2312(96)00048-3.

10. Tandel GS, Biswas M, Kakde OG, et al. A review on a deep learning perspective in brain cancer classification. *Cancers (Basel)*. 2019;11(1). doi:10.3390/cancers11010111.

11. Moeskops P, Viergever MA, Mendrik AM, De Vries LS, Benders MJNL, Isgum I. Automatic Segmentation of MR Brain Images with a Convolutional Neural Network. *IEEE Trans Med Imaging*. 2016;35(5):1252-1261. doi:10.1109/TMI.2016.2548501.

12. Havaei M, Davy A, Warde-Farley D, et al. Brain tumor segmentation with Deep Neural Networks. *Med Image Anal*. 2017;35:18-31. doi:10.1016/j.media.2016.05.004.

13. Zhao X, Wu Y, Song G, Li Z, Zhang Y, Fan Y. A deep learning model integrating FCNNs and CRFs for brain tumor segmentation. *Med Image Anal*. 2018;43:98-111. doi:10.1016/j.media.2017.10.002.

14. AlBadawy EA, Saha A, Mazurowski MA. Deep learning for segmentation of brain tumors: Impact of cross-institutional training and testing: Impact. *Med Phys*. 2018;45(3):1150-1158. doi:10.1002/mp.12752.

15. Li H, Li A, Wang M. A novel end-to-end brain tumor segmentation method using improved fully convolutional networks. *Comput Biol Med*. 2019;108:150-160. doi:10.1016/j.compbiomed.2019.03.014.

16. Zou KH, Warfield SK, Bharatha A, et al. Statistical Validation of Image Segmentation Quality Based on a Spatial Overlap Index. *Acad Radiol*. 2004;11(2):178-189. doi:10.1016/S1076-6332(03)00671-8.

17. Zhong C, Han J, Borowsky A, Parvin B, Wang Y, Chang H. When machine vision meets

histology: A comparative evaluation of model architecture for classification of histology sections. *Med Image Anal*. 2017;35:530-543. doi:10.1016/j.media.2016.08.010.

18. Gordillo N, Montseny E, Sobrevilla P. State of the art survey on MRI brain tumor segmentation. *Magn Reson Imaging*. 2013;31(8):1426-1438. doi:10.1016/j.mri.2013.05.002.

19. Wadhwa A, Bhardwaj A, Singh Verma V. A review on brain tumor segmentation of MRI images. *Magn Reson Imaging*. 2019;61:247-259. doi:10.1016/j.mri.2019.05.043.

20. Li R, Zhang W, Suk H-I, et al. Deep learning based imaging data completion for improved brain disease diagnosis. In: *Lecture Notes in Computer Science (Including Subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*. Vol 8675 LNCS. ; 2014:305-312. doi:10.1007/978-3-319-10443-0_39.

21. Hollon TC, Pandian B, Adapa AR, et al. Near real-time intraoperative brain tumor diagnosis using stimulated Raman histology and deep neural networks. *Nat Med*. 2020;26(1):52-58. doi:10.1038/s41591-019-0715-9.

22. Orringer DA, Pandian B, Niknafs YS, et al. Rapid intraoperative histology of unprocessed surgical specimens via fibre-laser-based stimulated Raman scattering microscopy. *Nat Biomed Eng*. 2017;1(2). doi:10.1038/s41551-016-0027.

23. Metz CE. Basic principles of ROC analysis. *Semin Nucl Med*. 1978;8(4):283-298. doi:10.1016/S0001-2998(78)80014-2.

24. Hanley JA, McNeil BJ. The meaning and use of the area under a receiver operating characteristic (ROC) curve. *Radiology*. 1982;143(1):29-36. doi:10.1148/radiology.143.1.7063747.

25. Schaumberg AJ, Rubin MA, Fuchs TJ. H&E-stained Whole Slide Deep Learning Predicts

SPOP Mutation State in Prostate Cancer. *bioRxiv*. 2016:64279. doi:10.1101/064279.

26. Wang D, Khosla A, Gargeya R, Irshad H, Beck AH. Deep Learning for Identifying Metastatic Breast Cancer. In: *International Symposium on Biomedical Imaging*. Cambridge; 2016.

27. Litjens G, Sánchez CI, Timofeeva N, et al. Deep learning as a tool for increased accuracy and efficiency of histopathological diagnosis. *Sci Rep*. 2016;6. doi:10.1038/srep26286.

28. Xu J, Luo X, Wang G, Gilmore H, Madabhushi A. A Deep Convolutional Neural Network for segmenting and classifying epithelial and stromal regions in histopathological images. *Neurocomputing*. 2016;191:214-223. doi:10.1016/j.neucom.2016.01.034.

29. Davis J, Goadrich M. The Relationship Between Precision-Recall and ROC Curves. In: *ICML 2006 - Proceedings of the 23rd International Conference on Machine Learning*. Vol 2006. ; 2006:233-240.

30. Raghavan V, Bollmann P, Jung GS. A Critical Investigation of Recall and Precision as Measures of Retrieval System Performance. *ACM Trans Inf Syst*. 1989;7(3):205-229. doi:10.1145/65943.65945.

31. Saito T, Rehmsmeier M. The Precision-Recall Plot Is More Informative than the ROC Plot When Evaluating Binary Classifiers on Imbalanced Datasets. *PLoS One*. 2015;10(3):1-21. doi:10.1371/journal.pone.0118432.

32. Li DC, Liu CW, Hu SC. A learning method for the class imbalance problem with medical data sets. *Comput Biol Med*. 2010;40(5):509-518. doi:10.1016/j.compbiomed.2010.03.005.

33. Li W, Liu K, Yan L, Cheng F, Lv YQ, Zhang LZ. FRD-CNN: Object detection based on small-scale convolutional neural networks and feature reuse. *Sci Rep*. 2019;9(1). doi:10.1038/s41598-019-52580-0.

34. Russakovsky O, Deng J, Su H, et al. ImageNet Large Scale Visual Recognition Challenge. *Int J Comput Vis*. 2015;115(3):211-252. doi:10.1007/s11263-015-0816-y.

35. Beitzel SM, Jensen EC, Frieder O. *Encyclopedia of Database Systems*. Boston, MA: Springer; 2009.

36. Lin T-Y, Maire M, Belongie S, et al. Microsoft COCO: Common Objects in Context. *Proc IEEE Comput Soc Conf Comput Vis Pattern Recognit*. 2015:3686-3693. doi:10.1109/CVPR.2014.471.

37. Fukushima K. Neocognitron: A self-organizing neural network model for a mechanism of pattern recognition unaffected by shift in position. *Biol Cybern*. 1980;36(4):193-202. doi:10.1007/BF00344251.

38. Lecun Y, Bengio Y, Hinton G. Deep learning. *Nature*. 2015;521(7553):436-444. doi:10.1038/nature14539.

39. Bottou L, Bousquet O. The Tradeoffs of Large Scale Learning. In: *Advances in Neural Information Processing Systems 20 - Proceedings of the 2007 Conference*. ; 2009. doi:10.7551/mitpress/8996.003.0015.

40. Menon MM, Heinemann KG. Classification of patterns using a self-organizing neural network. *Neural Networks*. 1988;1(3):201-215. doi:10.1016/0893-6080(88)90026-3.

41. Fukushima K, Wake N. Handwritten Alphanumeric Character Recognition by the Neocognitron. *IEEE Trans Neural Networks*. 1991;2(3):355-365. doi:10.1109/72.97912.

42. Lo SCB, Lou SLA, Chien M V., Mun SK. Artificial Convolution Neural Network Techniques and Applications for Lung Nodule Detection. *IEEE Trans Med Imaging*. 1995;14(4):711-718. doi:10.1109/42.476112.

43. Shih-Chung B Lo, Freedman MT, Lin JS, Mun SK. Automatic Lung Nodule Detection

using Profile Matching and Back-Propagation Neural Network Techniques. *J Digit Imaging*. 1993;6(1):48-54. doi:10.1007/BF03168418.

44. Ning F, Delhomme D, LeCun Y, Piano F, Bottou L, Barbano PE. Toward automatic phenotyping of developing embryos from videos. *IEEE Trans Image Process*. 2005;14(9):1360-1371. doi:10.1109/TIP.2005.852470.

45. Turaga SC, Murray JF, Jain V, et al. Convolutional Networks Can Learn to Generate Affinity Graphs for Image Segmentation. *Neural Comput*. 2010;22(2):511-538. doi:10.1162/neco.2009.10-08-881.

46. Deng J, Berg A, Satheesh S, Su H, Khosla A, Li F-F. ImageNet Large Scale Visual Recognition Challenge 2012 (ILSVRC2012). ImageNet.

47. Krizhevsky A, Sutskever I, Hinton GE. ImageNet Classification with Deep Convolutional Neural Networks. *Proc 25th Int Conf Neural Inf Process Syst*. 2012;3(9):322. doi:10.1007/s13398-014-0173-7.2.

48. Cireşan DC, Giusti A, Gambardella LM, Schmidhuber J. Mitosis detection in breast cancer histology images. *Int Conf Med Image Comput Comput Interv Springer, Berlin, Heidelb*. 2013:411-418. doi:10.1007/978-3-642-40763-5_51.

49. Cruz-Roa A, Basavanhally A, González F, et al. Automatic detection of invasive ductal carcinoma in whole slide images with convolutional neural networks. In: *Medical Imaging 2014: Digital Pathology*. Vol 9041. ; 2014:904103. doi:10.1117/12.2043872.

50. Ronneberger O, Fischer P, Brox T. U-net: Convolutional networks for biomedical image segmentation. In: *Lecture Notes in Computer Science (Including Subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*. Vol 9351. ; 2015:234-241. doi:10.1007/978-3-319-24574-4_28.

51. Long J, Shelhamer E, Darrell T. Fully Convolutional Networks for Semantic Segmentation. In: *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*. Vol 07-12-June. ; 2015:3431-3440. doi:10.1109/CVPR.2015.7298965.

52. Milletari F, Navab N, Ahmadi S-A. V-Net: Fully convolutional neural networks for volumetric medical image segmentation. In: *Proceedings - 2016 4th International Conference on 3D Vision, 3DV 2016*. ; 2016:565-571. doi:10.1109/3DV.2016.79.

53. Esteva A, Robicquet A, Ramsundar B, et al. A guide to deep learning in healthcare. *Nat Med*. 2019;25(1):24-29. doi:10.1038/s41591-018-0316-z.

54. Ching T, Himmelstein DS, Beaulieu-Jones BK, et al. Opportunities and obstacles for deep learning in biology and medicine. *J R Soc Interface*. 2018;15(141). doi:10.1098/rsif.2017.0387.

55. Shvets AA, Rakhlin A, Kalinin AA, Iglovikov VI. Automatic Instrument Segmentation in Robot-Assisted Surgery using Deep Learning. In: *Proceedings - 17th IEEE International Conference on Machine Learning and Applications, ICMLA 2018*. ; 2018:624-628. doi:10.1109/ICMLA.2018.00100.

56. Çiçek Ö, Abdulkadir A, Lienkamp SS, Brox T, Ronneberger O. 3D U-Net: Learning Dense Volumetric Segmentation from Sparse Annotation BT - Medical Image Computing and Computer-Assisted Intervention – MICCAI 2016. In: *International Conference on Medical Image Computing and Computer-Assisted Intervention*. ; 2016:424-432.

57. Liu Y, Gadepalli K, Norouzi M, et al. *Detecting Cancer Metastases on Gigapixel Pathology Images*. Mountain View; 2017.

58. Mercan C, Balkenhol M, van der Laak J, Ciompi F. From Point Annotations to Epithelial Cell Detection in Breast Cancer Histopathology using RetinaNet. In: *Medical Imaging with*

Deep Learning 2019. ; 2019.

59. Lu L, Daigle B. Prognostic Analysis of Histopathological Images Using Pre-Trained Convolutional Neural Networks. *bioRxiv Bioinforma*. 2019:1-17. doi:10.1101/620773.

60. Schaer R, Otálora S, Jimenez-del-Toro O, Atzori M, Müller H. Deep learning-based retrieval system for gigapixel histopathology cases and the open access literature. *J Pathol Inform*. 2019;10(1):19. doi:10.4103/jpi.jpi_88_18.

61. Fischer AH, Jacobson KA, Rose J, Zeller R. Hematoxylin and eosin staining of tissue and cell sections. *Cold Spring Harb Protoc*. 2008;3(5). doi:10.1101/pdb.prot4986.

62. Specify polygonal region of interest (ROI) - MATLAB roipoly. MathWorks Documentation. https://www.mathworks.com/help/images/ref/roipoly.html. Published 2020. Accessed February 20, 2020.

63. Shin HC, Roth HR, Gao M, et al. Deep Convolutional Neural Networks for Computer-Aided Detection: CNN Architectures, Dataset Characteristics and Transfer Learning. *IEEE Trans Med Imaging*. 2016;35(5):1285-1298. doi:10.1109/TMI.2016.2528162.

64. Abdolmanafi A, Duong L, Dahdah N, Adib IR, Cheriet F. Characterization of coronary artery pathological formations from OCT imaging using deep learning. *Biomed Opt Express*. 2018;9(10):4936. doi:10.1364/BOE.9.004936.

65. Nair V, Hinton GE. Rectified linear units improve Restricted Boltzmann machines. In: *ICML 2010 - Proceedings, 27th International Conference on Machine Learning*. ; 2010:807-814.

66. Glorot X, Bordes A, Bengio Y. Deep sparse rectifier neural networks. In: *Journal of Machine Learning Research*. Vol 15. ; 2011:315-323.

67. Radford A, Metz L, Chintala S. Unsupervised representation learning with deep

convolutional generative adversarial networks. In: *4th International Conference on Learning Representations, ICLR 2016 - Conference Track Proceedings*. ; 2016.

68. Maas AL, Hannun AY, Ng AY. Rectifier nonlinearities improve neural network acoustic models. In: *In ICML Workshop on Deep Learning for Audio, Speech and Language Processing*. ; 2013.

69. Bridle JS. Training Stochastic Model Recognition Algorithms as Networks can Lead to Maximum Mutual Information Estimation of Parameters. *Adv Neural Inf Process Syst*. 1990;(Ml):211-217.

70. Ioffe S, Szegedy C. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In: *32nd International Conference on Machine Learning, ICML 2015*. Vol 1. ; 2015:448-456.

71. Srivastava N, Hinton G, Krizhevsky A, Sutskever I, Salakhutdinov R. Dropout: A simple way to prevent neural networks from overfitting. *J Mach Learn Res*. 2014;15:1929-1958.

72. Zhang M, Zhang C, Wu X, et al. A neural network approach to segment brain blood vessels in digital subtraction angiography. *Comput Methods Programs Biomed*. 2020;185. doi:10.1016/j.cmpb.2019.105159.

73. Hou L, Samaras D, Kurc TM, Gao Y, Davis JE, Saltz JH. Patch-Based Convolutional Neural Network for Whole Slide Tissue Image Classification. In: *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*. Vol 2016-Decem. ; 2016:2424-2433. doi:10.1109/CVPR.2016.266.

74. Abadi M, Agarwal A, Barham P, et al. *TensorFlow: Large-Scale Machine Learning on Heterogeneous Distributed Systems*. Mountain View; 2015.

75. Cireşan DC, Giusti A, Gambardella LM, Schmidhuber J. Deep neural networks segment

neuronal membranes in electron microscopy images. In: *Advances in Neural Information Processing Systems*. Vol 4. ; 2012:2843-2851.

76. Murphy K. *Machine Learning: A Probabilistic Perspective*. Cambridge: MIT Press; 2013.

77. Bishop CM. *Pattern Recognition and Machine Learning*. New York: Springer; 2016.

78. Kingma DP, Ba JL. Adam: A method for stochastic optimization. In: *3rd International Conference on Learning Representations, ICLR 2015 - Conference Track Proceedings*. ; 2015.

79. Li R, Johansen JS, Ahmed H, et al. *Training on the Test Set? An Analysis of Spampinato et Al. [31]*. West Lafayette; 2018.

80. Mamun-Ur-Rashid Khan M. Numerical Integration Schemes for Unequal Data Spacing. *Am J Appl Math*. 2017;5(2):48. doi:10.11648/j.ajam.20170502.12.

81. Montgomery DC, Runger GC, Hubele NF. *Engineering Statistics*. 5th ed. Wiley; 2012.

82. Hansen LK, Salamon P. Neural Network Ensembles. *IEEE Trans Pattern Anal Mach Intell*. 1990;12(10):993-1001. doi:10.1109/34.58871.

83. Perrone MP, Cooper LN. When networks disagree: Ensemble methods for hybrid neural networks. In: ; 1995:342-358. doi:10.1142/9789812795885_0025.

84. Krogh A, Vedelsby J. Neural Network Ensembles, Cross Validation, and Active Learning. *Adv Neural Inf Process Syst 7*. 1995:231–238. doi:10.1.1.37.8876.

85. Goodenough DJ, Rossmann K, Lusted LB. Radiographic applications of receiver operating characteristic (ROC) curves. *Radiology*. 1974;110(1):89-95. doi:10.1148/110.1.89.

86. Pitas I, Venetsanopoulos AN. *Nonlinear Digital Filters: Principles and Applications*. Kluwer Academic Publishers; 1990.

87. Hwu W. *GPU Computing Gems*. Elsevier; 2011.

88. Tan L, Jiang J. *Digital Signal Processing: Fundamentals and Applications*. Third. Academic Press; 2019.

89. Blackledge JM. *Digital Image Processing: Mathematical and Computational Methods*. Loughborough University: Woodhead Publishing; 2006.

90. Alanis AY, Arana-Daniel N, López-Franco C. *Artificial Neural Networks for Engineering Applications*. Guadalajara: Elsevier; 2019.

91. 2 - D median filtering - MATLAB medfilt2. MathWorks Documentation. https://www.mathworks.com/help/images/ref/medfilt2.html. Published 2020. Accessed March 2, 2020.

92. 2 - D gaussian filtering of images - MATLAB imgaussfilt. MathWorks Documentation. https://www.mathworks.com/help/images/apply-gaussian-smoothing-filters-to-images.html. Published 2020. Accessed March 2, 2020.

93. Wickham H. *ggplot2: Elegant Graphics for Data Analysis*. New York; 2016.

94. Rezatofighi H, Tsoi N, Gwak J, Sadeghian A, Reid I, Savarese S. Generalized intersection over union: A metric and a loss for bounding box regression. In: *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*. Vol 2019-June. ; 2019:658-666. doi:10.1109/CVPR.2019.00075.

95. Dice LR. Measures of the Amount of Ecologic Association Between Species. *Ecology*. 1945;26(3):297-302. doi:10.2307/1932409.

96. Cordts M, Omran M, Ramos S, et al. The Cityscapes Dataset for Semantic Urban Scene Understanding. In: *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*. Vol 2016-Decem. ; 2016:3213-3223. doi:10.1109/CVPR.2016.350.

97. Abu Alhaija H, Mustikovela SK, Mescheder L, Geiger A, Rother C. Augmented Reality Meets Computer Vision: Efficient Data Generation for Urban Driving Scenes. *Int J Comput Vis*. 2018;126(9):961-972. doi:10.1007/s11263-018-1070-x.

98. Weller DS, Ramani S, Nielsen JF, Fessler JA. Monte Carlo SURE-based parameter selection for parallel magnetic resonance imaging reconstruction. *Magn Reson Med*. 2014;71(5):1760-1770. doi:10.1002/mrm.24840.

99. Everingham M, Eslami SMA, Van Gool L, Williams CKI, Winn J, Zisserman A. The Pascal Visual Object Classes Challenge: A Retrospective. *Int J Comput Vis*. 2015;111(1):98-136. doi:10.1007/s11263-014-0733-5.

100. Mohsen H, El-Dahshan E-SA, El-Horbaty E-SM, Salem A-BM. Classification using deep learning neural networks for brain tumors. *Futur Comput Informatics J*. 2018;3(1):68-71. doi:10.1016/j.fcij.2017.12.001.

101. Weinstein JN, Collisson EA, Mills GB, et al. The Cancer Genome Atlas Pan-Cancer Analysis Project. *Nat Genet*. 2013;45(10):1113-1120. doi:10.1038/ng.2764.

102. Coons SW, Johnson PC, Scheithauer BW, Yates AJ, Pearl DK. Improving diagnostic accuracy and interobserver concordance in the classification and grading of primary gliomas. *Cancer*. 1997;79(7):1381-1393. doi:10.1002/(SICI)1097-0142(19970401)79:7<1381::AID-CNCR16>3.0.CO;2-W.

103. Daumas-Duport C, Scheithauer B, O'Fallon J, Kelly P. Grading of astrocytomas: A simple and reproducible method. *Cancer*. 1988;62(10):2152-2165. doi:10.1002/1097-0142(19881115)62:10<2152::AID-CNCR2820621015>3.0.CO;2-T.

104. Weller RO. Grading of brain tumours. The British experience. *Neurosurg Rev*. 1992;15(1):7-11. doi:10.1007/BF02352059.

105. Gupta M, Djalilvand A, Brat DJ. Clarifying the diffuse gliomas: An update on the morphologic features and markers that discriminate oligodendroglioma from astrocytoma. *Am J Clin Pathol*. 2005;124(5):755-768. doi:10.1309/6JNX4PA60TQ5U5VG.

106. Li C, Chen J, Qin G. Partial Youden index and its inferences. *J Biopharm Stat*. 2019;29(2):385-399. doi:10.1080/10543406.2018.1535502.

107. Youden WJ. Index for rating diagnostic tests. *Cancer*. 1950;3(1):32-35. doi:10.1002/1097-0142(1950)3:1<32::AID-CNCR2820030106>3.0.CO;2-3.

108. Schisterman EF, Perkins NJ, Liu A, Bondell H. Optimal cut-point and its corresponding Youden index to discriminate individuals using pooled blood samples. *Epidemiology*. 2005;16(1):73-81. doi:10.1097/01.ede.0000147512.81966.ba.

109. Han SS, Park GH, Lim W, et al. Deep neural networks show an equivalent and often superior performance to dermatologists in onychomycosis diagnosis: Automatic construction of onychomycosis datasets by region-based convolutional deep neural network. *PLoS One*. 2018;13(1). doi:10.1371/journal.pone.0191493.

110. Crivellaro C, Landoni C, Elisei F, et al. Combining positron emission tomography/ computed tomography, radiomics, and sentinel lymph node mapping for nodal staging of endometrial cancer patients. *Int J Gynecol Cancer*. 2020;30(3).

111. Ruifrok AC, Johnston DA. Quantification of histochemical staining by color deconvolution. *Anal Quant Cytol Histol*. 2001;23(4):291-299.

112. Bengio Y. Deep Learning of Representations for Unsupervised and Transfer Learning. In: *JMLR: Workshop and Conference Proceedings*. Vol 7. ; 2011:1-20.

113. Yosinski J, Clune J, Bengio Y, Lipson H. How transferable are features in deep neural networks? In: *Advances in Neural Information Processing Systems*. Vol 4. ; 2014:3320-

3328.

114. Bayramoglu N, Heikkilä J. Transfer learning for cell nuclei classification in histopathology images. In: *Lecture Notes in Computer Science (Including Subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*. Vol 9915 LNCS. ; 2016:532-539. doi:10.1007/978-3-319-49409-8_46.

115. Krizhevsky A. *Learning Multiple Layers of Features from Tiny Images*. Toronto; 2009. doi:10.1.1.222.9220.

116. Hope T, Resheff YS, Lieder I. *Learning Tensorflow: A Guide to Building Deep Learning Systems*. Sebastopol: O'Reilly; 2017.

117. Szegedy C, Liu W, Jia Y, et al. Going deeper with convolutions. In: *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*. Vol 07-12-June. ; 2015:1-9. doi:10.1109/CVPR.2015.7298594.

118. He K, Zhang X, Ren S, Sun J. Deep Residual Learning for Image Recognition. In: *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*. ; 2015:770-778. doi:10.1109/CVPR.2016.90.

119. Bengio Y, Simard P, Frasconi P. Learning Long-Term Dependencies with Gradient Descent is Difficult. *IEEE Trans Neural Networks*. 1994;5(2):157-166. doi:10.1109/72.279181.

120. Wolpert DH. Stacked generalization. *Neural Networks*. 1992;5(2):241-259. doi:10.1016/S0893-6080(05)80023-1.

7. Appendix

7.1. Material Contained in CSV Files

Our code wrote the following quantities to CSV files during evaluation over the validation set:

- The number of epochs performed at the time of evaluation
- The number of total patches in the training set before and after balancing▲
- The number of normal patches in the training set before and after balancing▲
- The number of tumor patches in the training set▲·
- The ratio between the number of normal and tumor patches in the training set after balancing▲
- The proportion of all patches in the training set remaining after balancing▲
- The proportion of normal patches in the training set remaining after balancing▲
- The fraction of the total number of patches in the training and validation sets comprised by the training set after balancing▲
- The number of total patches in the validation set▲
- The number of total patches in the portion of the validation set provided to the model
- The number of normal patches in the provided portion of the validation set
- The number of tumor patches in the provided portion of the validation set
- The number of true positives incurred during validation
- The number of false positives incurred during validation
- The number of true negatives incurred during validation
- The number of false negatives incurred during validation

- The number of patches correctly identified during validation
- The overall accuracy of the model on the provided portion of the validation set
- The sensitivity of the model on the provided portion of the validation set
- The specificity of the model on the provided portion of the validation set
- The false positive rate of the model on the provided portion of the validation set

▲: These values were properties of the training and validation sets and therefore did not change with the number of training epochs performed.

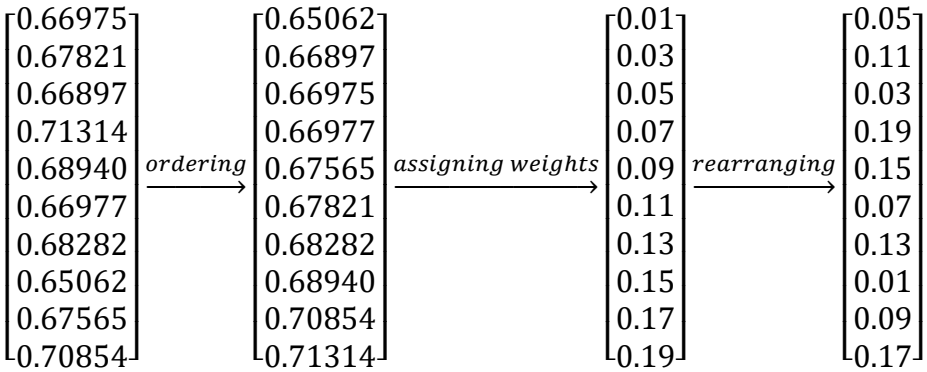
•: Since the training set balancing step only removed normal patches in the preliminary training set, the number of tumor patches in the training set remained unchanged after balancing.

7.2. Weighted Heatmap Averaging

We developed two methods for performing weighted heatmap averaging. The former of these assigned linearly-spaced weights to the probability heatmaps based on their associated key classifier metric values. Conversely, the latter approach used the key classifier metric value of each probability heatmap to compute a corresponding weight value. In both cases, we made certain to require the weights to sum to one.

7.2.1. Linear Method

The weights in this approach spanned from 0.01 to 0.19 in increments of 0.02. We sorted the probability heatmaps in order of increasing key classifier metric values and assigned the weights accordingly. An example of this process for Image 004 in the AUC study with no filtering is presented on the following page.



The main benefit of this approach was that it did not require the calculation of unique weights for all probability heatmaps; however, it did include two ordering steps which added further complexity to our system. Likewise, we found this method to produce weights that failed to fit the distributions of AUC values as closely as its counterpart, likely as a result of the standardized weight values. Conversely, we observed the opposite trend in the MAP study.

7.2.2. Normalization Method

Unlike the previous method, this approach computed the weight values using the corresponding key classifier metric values. These steps are presented below:

1) Identify the minimum key classifier metric value.
2) Subtract this minimum value from each key classifier metric value to obtain an associated deviation value.
3) Add a small offset to each deviation value to obtain a corresponding modified deviation value. This offset ensured that our methodology did not assign a weight of zero to the probability heatmap with the lowest key classifier metric value, therefore excluding it. Our choice of offset was 0.01, an arbitrary selection based on the general scale of the deviation values.

4) Calculate the total deviation value by summing all modified deviation values.
5) Divide each modified deviation value by the total deviation value to calculate the weight for each probability heatmap.

Since this process involved several steps and may be rather confusing to follow, we demonstrate this procedure on the same AUC values discussed previously below and on the following page.

$$AUC = \begin{bmatrix} 0.66975 \\ 0.67821 \\ 0.66897 \\ 0.71314 \\ 0.68940 \\ 0.66977 \\ 0.68282 \\ 0.65062 \\ 0.67565 \\ 0.70854 \end{bmatrix}$$

$$offset = 0.01$$

1) $$minimum = \min(AUC) = \min\left(\begin{bmatrix} 0.66975 \\ 0.67821 \\ 0.66897 \\ 0.71314 \\ 0.68940 \\ 0.66977 \\ 0.68282 \\ 0.65062 \\ 0.67565 \\ 0.70854 \end{bmatrix}\right) = 0.65062$$

2) $deviation = AUC - minimum = \begin{bmatrix} 0.66975 \\ 0.67821 \\ 0.66897 \\ 0.71314 \\ 0.68940 \\ 0.66977 \\ 0.68282 \\ 0.65062 \\ 0.67565 \\ 0.70854 \end{bmatrix} - 0.65062 = \begin{bmatrix} 0.01913 \\ 0.02759 \\ 0.01835 \\ 0.06252 \\ 0.03878 \\ 0.01915 \\ 0.03220 \\ 0.00000 \\ 0.02503 \\ 0.05792 \end{bmatrix}$

3) $deviation_{modified} = deviation + offset = \begin{bmatrix} 0.01913 \\ 0.02759 \\ 0.01835 \\ 0.06252 \\ 0.03878 \\ 0.01915 \\ 0.03220 \\ 0.00000 \\ 0.02503 \\ 0.05792 \end{bmatrix} + 0.01 = \begin{bmatrix} 0.02913 \\ 0.03759 \\ 0.02835 \\ 0.07252 \\ 0.04878 \\ 0.02915 \\ 0.04220 \\ 0.01000 \\ 0.03503 \\ 0.06792 \end{bmatrix}$

4) $deviation_{total} = \text{sum}\left(deviation_{modified}\right) = \text{sum}\left(\begin{bmatrix} 0.02913 \\ 0.03759 \\ 0.02835 \\ 0.07252 \\ 0.04878 \\ 0.02915 \\ 0.04220 \\ 0.01000 \\ 0.03503 \\ 0.06792 \end{bmatrix}\right) = 0.40067$

5) $weights = deviation_{modified}/deviation_{total} = \begin{bmatrix} 0.02913 \\ 0.03759 \\ 0.02835 \\ 0.07252 \\ 0.04878 \\ 0.02915 \\ 0.04220 \\ 0.01000 \\ 0.03503 \\ 0.06792 \end{bmatrix} /0.40067 = \begin{bmatrix} 0.07270 \\ 0.09382 \\ 0.07075 \\ 0.18100 \\ 0.12175 \\ 0.07275 \\ 0.10534 \\ 0.02496 \\ 0.08743 \\ 0.16951 \end{bmatrix}$

This approach allowed probability heatmaps with superior capabilities to contribute more to heatmap averaging, while also allowing for representation from all heatmaps. Likewise, this completely omitted the ordering step required in the previous analysis. Furthermore, this method generated weights which tightly fit the distributions of AUC values. Unfortunately, it did not perform as well in the MAP study. It also did not include standardized weight values, which could have entailed some difficulties when comparing weighted heatmap averaging over distinct testing images.