UNIVERSITY OF CALIFORNIA SAN DIEGO

# A Modular Framework for Preprocessing and Interpretable Multi-Instance Learning in Cancer Whole Slide Image Analysis

A Thesis submitted in partial satisfaction of the requirements
for the degree Master of Science

in

Bioengineering

by

Lorenzo Olmo Marchal

Committee in charge:

Professor Ludmil B. Alexandrov, Chair
Professor Benjamin Smarr
Professor Vineet Bafna

2025

THESIS APPROVAL PAGE

The Thesis of Lorenzo Olmo Marchal is approved, and it is acceptable in quality and form for publication on microfilm and electronically.

University of California San Diego

2025

# DEDICATION

With great gratitude to my family, friends, mentors, and colleagues for all their support and encouragement. Without you, I would not be who I am today. I would like to especially recognize my mentors— Dr. Erik N. Bergstrom, Dr. Ludmil B. Alexandrov, and Dr. Yu “Max” Qian— for their invaluable guidance and continued mentorship, which made this thesis possible.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES, EQUATIONS & ALGORITHMS

LIST OF ABBREVIATIONS

| | |
|---|---|
| WSI | Whole Slide Image |
| H&E | Hematoxylin and eosin |
| MPP | Microns per pixel |
| MIL | Multi Instance Learning |
| CNN | Convolutional Neural Network |
| abMIL | Attention-based Multi Instance Learning (i.e., Attention-pooled MIL) |
| AI | Artificial Intelligence |
| MSI | Microsatellite Instability |
| MSS | Microsatellite Stable |
| B-ALL | B-cell Acute Lymphoblastic Leukemia |
| FCM | Flow Cytometry |

ACKNOWLEDGEMENTS

My sincere gratitude goes to my advisor, Professor Ludmil B. Alexandrov, for his invaluable mentorship and for providing the support and computational resources necessary for this thesis.

I would also like to thank the computational pathology team at the Alexandrov lab. I am especially grateful to Dr. Erik Bergstrom for his direct supervision and detailed feedback on Chapters 1 and 2, and to Banghua Xu for his help in testing the SlideLab pipeline.

I thank my committee members, Professor Ludmil B. Alexandrov, Professor Benjamin Smarr, and Professor Vineet Bafna, for their time and thoughtful guidance.

## VITA

2021-2024 Bachelor of Science in Bioengineering: Bioinformatics, University of California San Diego

2024-2025 Master of Science in Bioengineering, University of California San Diego

## FIELD OF STUDY

Major Field: Computational biology
Studies in Bioinformatics and computational pathology
Professor Ludmil B. Alexandrov

ABSTRACT OF THE THESIS

A Modular Framework for Preprocessing and Interpretable Multi-Instance Learning in Cancer Whole Slide Image Analysis

by

Lorenzo Olmo Marchal

Master of Science in Bioengineering
University of California San Diego, 2025

Professor Ludmil B. Alexandrov, Chair

Despite the potential of computational pathology, the current focus in the field on deep learning architectures has reduced advances in other essential areas. The literature consistently describes that the current largest hindrances towards advances in trustworthy computational pathology is unstandardized preprocessing and the “black box” nature of models, which limits reproducibility and clinical trust, respectively. This thesis addresses these challenges through the

development of two independent tools that can be used in conjunction or flexibly implemented in existing computational pathology workflows.

First, this work introduces SlideLab, a novel, modular, and efficient open-source framework for preprocessing whole slide images. Compared to current state-of-the-art tools, SlideLab produces higher-quality datasets with improved consistency. Benchmarking demonstrates that models trained on SlideLab-processed data show enhanced performance and generalizability in classifying microsatellite instability (MSI-H vs. MSS) in colorectal cancer.

Second, this thesis presents the Polarized Attentional Certainty (PAC) metric, a new interpretability score for attention-based multi-instance learning models. PAC fuses predictive certainty and attentional focus into a unified score to reveal a model's decision-making process. Its utility is demonstrated in two separate case studies:  in histopathology, PAC analysis confirmed that a model correctly associated Microsatellite Instability-High (MSI-H) status with an inflamed tumor microenvironment. In flow cytometry, PAC revealed that a model learned to define B-cell Acute Lymphoblastic Leukemia (B-ALL) by its clinical CD10+/CD34+ blast immunophenotype, clearly distinguishing it from other mature leukocyte populations.

Together, SlideLab and PAC provide an integrated framework that enhances both the practical foundation and the analytical understanding of deep learning in computational pathology, paving the way for more robust, transparent, and trustworthy AI-driven tools.

# INTRODUCTION

## Transition from Traditional to Computational Pathology

Histopathology has long been the gold standard for disease diagnosis, prognosis, and therapeutic monitoring, relying on pathologists' visual examination of stained tissue samples under a microscope. While this manual approach has been effective for decades, it faces significant limitations in subjectivity, labor intensity and scalability [1]. With the growing diagnostic demands of precision medicine and large-scale biomedical research, these constraints have become increasingly problematic by creating critical bottlenecks in throughput [1][2][3]

In response to these limitations, computational pathology has emerged as a rapidly advancing subspecialty in the last two decades; focused on the quantitative, large-scale analysis of pathological data like Whole Slide Images (WSI) [3]. As high-resolution digitized representations of histology slides, WSI contain diagnostically relevant features that can be leveraged by computational methods such as deep learning [4][5]. Unlike conventional methods, which emphasize individual slide examinations, these computational approaches utilize their large-scale data analysis capabilities to identify subtle but pertinent features (e.g., nuclear atypia, cellular spatial relationships, tissue architecture alterations) that may be difficult to quantify consistently by human observers alone [4][5].

**The Computational Pathology Workflow**

A standard computational pathology workflow, as depicted in *Figure 0*, is composed of five key stages after data collection: (I) preprocessing, (II) model training, (III) model evaluation and validation, (IV) model interpretability, and (V) deployment in research or clinical settings [6][7].

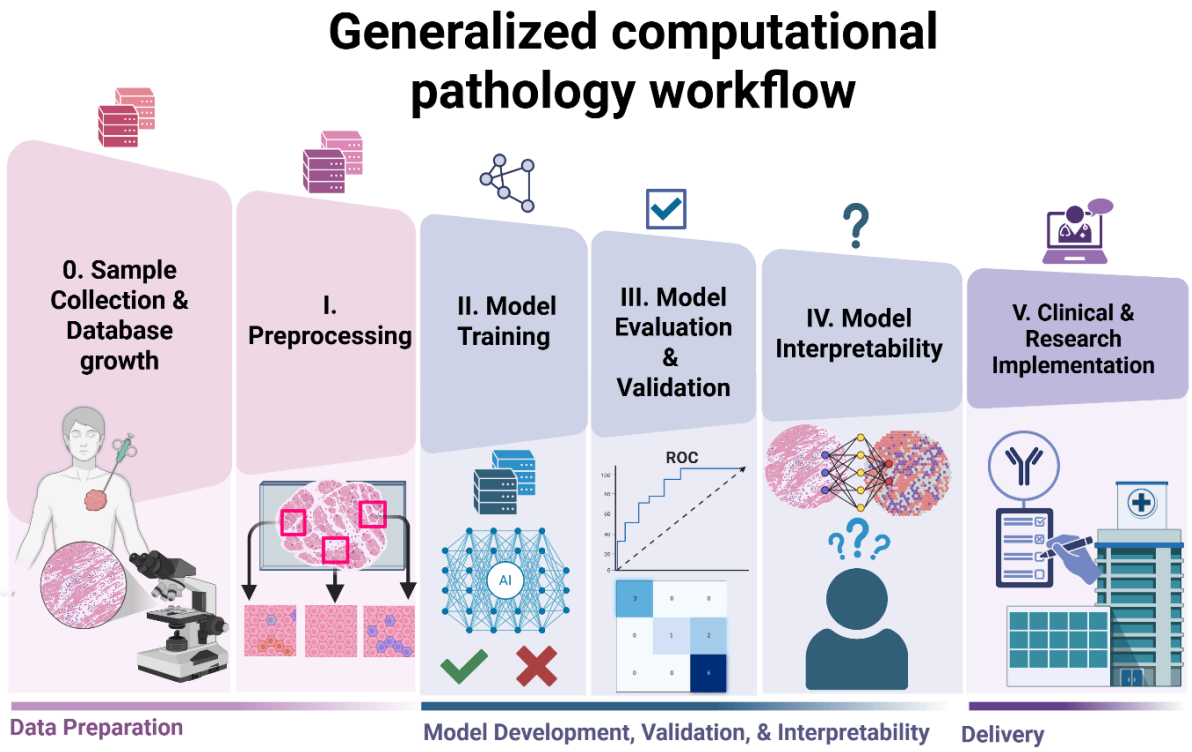


**Figure 0. A generalized workflow for computational pathology.**

*The process begins with (0) Sample collection, where pathological data is obtained and prepared for analysis. This is followed by (I) Preprocessing of the images, which includes highlighting and segmenting relevant features. The preprocessed data is then used in (II) Model Training, where an AI algorithm learns to classify or predict outcome. Following training, the model's performance is examined during (III) Model Evaluation and Validation, where model performance is tested on independent sets and metrics like ROC curves and confusion matrix are used to measure performance quality. In (IV) Model Interpretability, the AI model's decision-making process is analyzed to ensure valid reasoning. The final stage is (V) Clinical & Research implementation, where the validated and interpretable model is integrated into a hospital or research setting to assist with diagnosis/prognosis and treatment.*

Preprocessing, as the foundational first step, is critical for ensuring the quality and consistency of all subsequent analyses. This stage involves the extraction and standardization of relevant biological data from WSIs, which is done through techniques like tissue segmentation, stain normalization, and various quality control procedures [6][8]. During this process, WSIs are divided into smaller tissue patches for deep-learning models; typically, multi-instance learning frameworks or convolutional neural networks for tasks such as classification, regression, and segmentation [8]. Real world-word data is messy, and inconsistent or poorly preprocessed data can introduce noise and bias that directly affect all downstream analyses—such as poor model performance due to spurious correlations from artifacts [9]. Therefore, robust preprocessing is essential for a successful computational pathology protocol.

Following preprocessing, deep-learning models are optimized for a wide range of tasks. These models have demonstrated remarkable success in tasks ranging from tumor detection and grading to inferring molecular features (e.g., Homologous Recombination) directly from hematoxylin and eosin (H&E)-stained slides [10][11]. For instance, a pivotal early success in the field was the prediction of microsatellite instability (MSI) status in colorectal cancer [12]. In addition to prediction, these models have shown great performance in segmentation tasks such as nuclei and tissue compartment segmentation for the study of tumor microenvironments and morphometric quantification of WSI [13][14]. After training, models undergo rigorous evaluation and validation through independent validation cohorts to assess generalizability and robustness. If a model cannot perform reliably across diverse datasets, potential biases or limitations must be identified and addressed.

Prior to integration into clinical or research settings, interpretability techniques are crucial to understanding the model's decision-making process, thereby promoting transparency, trust,

and regulatory compliance. These techniques aim to find the features that are the most relevant towards model decisions, thus providing an explanation for model predictions and bypassing the "black box" nature of deep-learning models. Some widespread interpretability methods used in computational pathology include attention heatmaps or saliency maps, which visually highlight regions of an input image that a model gives more weight to when making a decision [15][16]. The application of these techniques is vital to ensure that model reasoning aligns with both clinical and biological rationale [7].

The final stage involves the delivery of the validated and interpretable models into research and clinical settings. This includes deploying the models into user-friendly platforms, ensuring seamless integration with existing laboratory information systems, and establishing continuous mechanisms for continuous monitoring and updates based on feedback [3][7].

**The Research Gap: The Need for Standardized Preprocessing and Interpretability**

In contrast to the rapid development of new models, preprocessing and interpretability methods in computational pathology remain either largely unexplored or unstandardized [17]. This oversight poses significant hurdles to the widespread adoption and reliable application of artificial intelligence in pathology. A 2023 survey of 70 pathologists revealed that while 87% expressed enthusiasm about integrating computational pathology into clinical workflows, they voiced concerns extending beyond model performance [18]. These included uncertainty about quality assessment , specifically citing issues such as variability in tissue staining and automated quality control measures, which are typically handled by pathologists prior to evaluating the tissue. Pathologists also voiced limited familiarity with the underlying algorithmic processes,

viewing computational pathology largely as a "black box" that they were hesitant to trust, especially given patient impact and potential liability issues [18][19][20].

From a technical standpoint, similar obstacles are well documented in existing literature [17][21][22]. One of the most cited obstacles in computational pathology is the lack of reproducibility, stemming from restricted, proprietary datasets and insufficient transparency regarding the specific preprocessing steps undertaken (e.g., patch selection criteria, stain normalization methods, image quality metrics) [17][21][22]. This lack of transparency makes it difficult to validate and trust individual results. Another issue is the limited interpretability of models. Due to the lack of annotations beyond slide-level, a large portion of models in computational pathology operate under weak or no supervision (e.g. Multi Instance Learning) [15][23]. While these less-supervised methods are powerful, they are inherently more challenging not only to validate but also interpret. In such models, it often remains unclear if they are learning causal, biologically relevant features or merely relying on spurious correlations within the data. This concern is especially pressing in computational pathology, where model predictions can have direct and profound implications for patient diagnosis, treatment decisions, and research direction. The inability to fully comprehend a model's reasoning in these high-stakes scenarios hinders the safe and ethical deployment of AI in clinical and research settings [3][20].

**Thesis Aims and Contributions**

This thesis aims to address these significant challenges in preprocessing and model interpretability within computational pathology. The core contributions of this work are first, the development of SlideLab, a novel and robust preprocessing module designed to optimize the

creation of high-quality WSI datasets; second, the introduction of a new interpretability method, polarized attentional certainty (PAC), with an exploration of its applications in attention-based models to enhance the transparency and trustworthiness of AI systems in pathology.

**Thesis Outline**

The remainder of this thesis is structured as follows. Chapter 1 focuses on preprocessing with a brief review of the existing landscape of preprocessing in computational pathology and presents the methodology for the SlideLab preprocessing module, detailing its architecture and implementation. Chapter 1 additionally includes the benchmarking of SlideLab against current state-of-the-art preprocessing modules in computational, quality control, and model performance metrics. Chapter 2 provides a brief detailed background on current available interpretability techniques of gold standard attention based multi-instance learning (abMIL) models for computational pathology. Furthermore, it introduces PAC, its methodology, and applications. To display the potential of PAC, two case studies where abMIL is applicable are presented: (I) the application of PAC to study the tumor microenvironment of colorectal Microsatellite Unstable versus that of Microsatellite Stable tumors according to model predictions; (II) the application of PAC to analyze how a model immunophenotypes cells to predict for B-cell Acute Lymphocytic Leukemia (B-ALL) using single-cell Flow Cytometry.

# Chapter 1 : SlideLab

## 1.1 Introduction: The Challenge of Whole Slide Image Processing

Unlike traditional pathology, which relies on visual inspection of individual histology slides by a pathologist, computational pathology leverages statistical analysis across numerous slides to identify distinguishing characteristics for diagnostic, prognostic, and segmentation tasks. Whole Slide Images (WSIs) —digitized histology images— are an invaluable resource in computational pathology. Their high-resolution digital format offers and preserves detailed cellular information, such as morphology and spatial organization, enabling more systematic and reproducible analysis compared to manual inspections.

Despite their potential, several challenges hinder the direct use of WSI in automated analyses. A primary obstacle is the immense computational cost associated with their gigapixel scale. With dimensions exceeding 100,000 × 100,000 pixels, WSIs can be extremely large, with files ranging from 640 MB to 3.7 GB even after compression [23]. This makes full-image utilization computationally infeasible for most downstream analyses like deep-learning, which are computationally intensive in their own right.

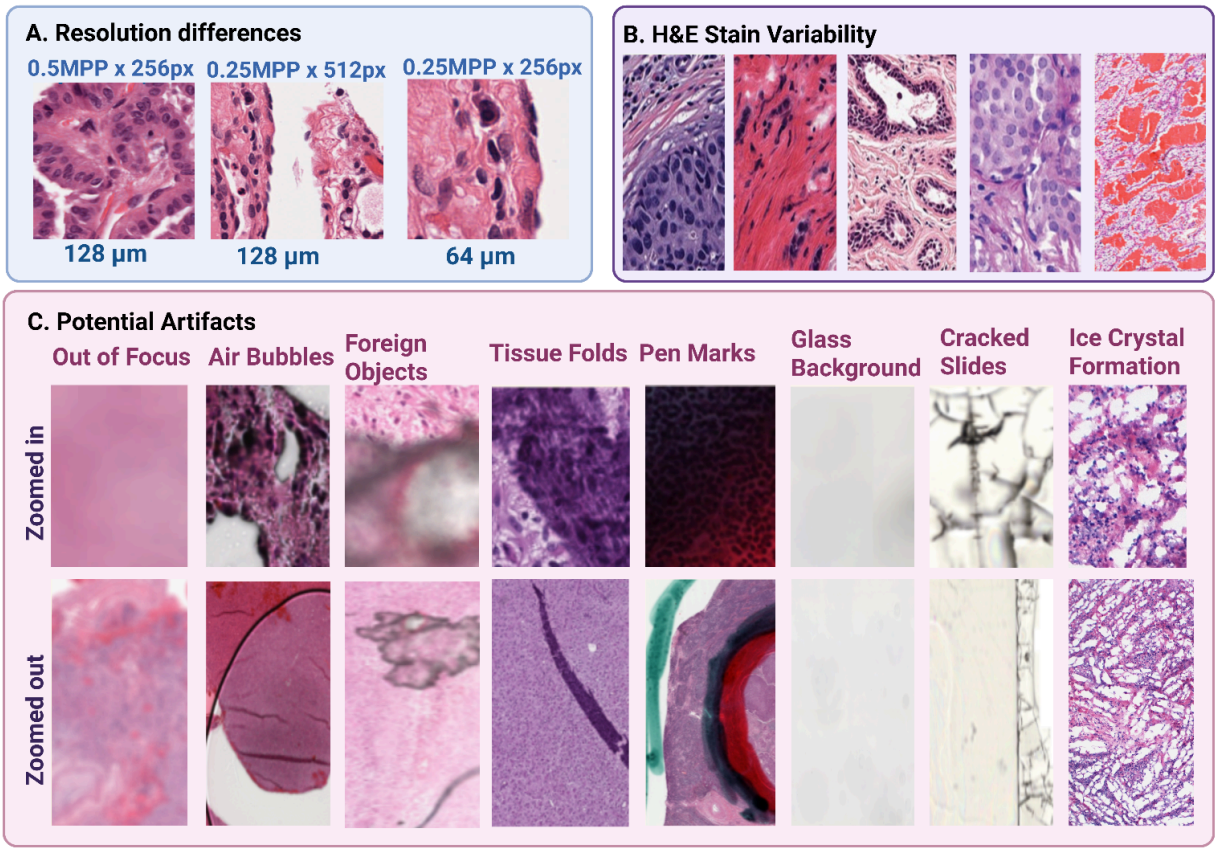


**Figure 1.1:** ***Sources of potential variability in Whole Slide Images.***

*(A) Resolution differences between slides and their effect on tissue extraction, (B) hematoxylin and Eosin variability, and (C) potential artifacts (i.e., non-biologically relevant sections) that can be found in Whole Slide Images.*

Beyond computational demands, variability in tissue preparation and digitization introduces further complexity. Manual slide preparation often introduces inconsistencies and artifacts that can severely impede accurate results in computational pathology workflows [8][24]. For example, Hematoxylin and Eosin (H&E) staining protocols vary across institutions (Figure 1.1B), creating stain-dependent biases that affect model generalizability [8][24]. WSIs may also contain non-biological features like the glass background or preparation artifacts such as pen marks, tissue folds, blurry regions, air bubbles, and foreign objects, all of which compromise data quality by introducing noise (Figure 1.1C) [25][26][27]. The tissue conservation further exacerbates variability [27] . As an example, ice crystal formation, which distorts the native

tissue architecture, is very common in flash-frozen (FF) slides in comparison to Formalin-fixed paraffin-embedded (FFPE) slides (Figure 1.1C) [28].

Further variability in WSI processing stems from the digital microscopes used for digitalization. WSIs are scanned at specific resolutions, typically defined by the objective lens magnification (e.g., 20x, 40x) or the microns-per-pixel (MPP). Effectively, these differences in resolution greatly impact the amount of microns captured during preprocessing. As pictured in Figure 1.1A, to extract the same amount of microns from a slide digitized at 0.25 MPP (40x) as one at a 0.5 MPP (20x) resolution, twice the amount of pixels must be captured [8]. These variations can greatly impact data standardization in large data cohorts, where slides come from different institutions and utilize different scanners.

Consequently, WSIs require a series of crucial preprocessing steps to ensure data quality and consistency before their effective application in downstream computational workflows. Previous studies have explored several approaches to improve WSI preprocessing, broadly categorized as slide-level and instance-level techniques.

## Existing Preprocessing Approaches

WSI patching is a fundamental instance-level technique introduced to address the high computational demands of processing gigapixel-scale images [8][29]. This method involves dividing WSIs into smaller, fixed-size regions (i.e., patches or instances) that are processed independently, allowing for more efficient computation and localized feature extraction. The optimal patch size needs to be adjusted based on the desired resolution, as depicted in Figure 1.1A. While patching significantly reduces the memory footprint of processing a slide,

processing tens of thousands of individual instances still presents a considerable computational cost.

To address these inefficacies, patching is often complemented by slide-level automatic thresholding methods. These techniques aim to identify and segment relevant tissue regions while excluding irrelevant background areas (e.g., the glass slide). Typically, WSIS are downscaled prior to applying methods like Otsu's adaptive thresholding to separate foreground tissue from background noise [30][31][32]. Similarly, contour [15][33] and canny edge detection algorithms are applied to define tissue boundaries and identify regions of interest [34][35]. Other algorithms, such as those for removing tissue folds as suggested by Kothari et al. (2013) or RGB channel thresholding for pen-marks , are also applied at this stage [36][37].

In addition to these slide-level approaches, supplementary quality control measures are often introduced at the instance-level following tissue segmentation. For example, blur detection methods such as Laplacian variance or Fast Fourier Transform–based approaches are commonly used to filter out out-of-focus patches and improve the reliability of downstream analyses [38][39].

While generally robust, these traditional computer vision algorithms can be unreliable when faced with slide quality issues or heterogeneity in staining. To overcome this limitation, there has been a growth of deep-learning models like GrandQC [26], HistoQC [27], and HistoROI [25]. These models are specifically trained to identify tissue and various artifacts, and despite their computational expense, they have shown to be more accurate and sensitive to hard-to-find artifacts like air bubbles. However, these newer methods still have their own limitations. For example, in solid-tumor applications, models like GrandQC [26] may misclassify adipose tissue as relevant tissue, despite typically being considered an artifact in these settings

[27][40]. By integrating both traditional and deep-learning approaches, computational pipelines can effectively reduce unnecessary computation on non-tissue areas and improve the overall quality and consistency of the data used for downstream analysis.

In terms of handling H&E staining variability, a crucial instance-level step to improve the overall quality and consistency of the data used for downstream analysis is stain normalization. This technique standardizes the color and intensity of digital pathology slides, mitigating inconsistencies caused by different labs, reagents, and scanners. Traditional algorithms such as Reinhard [41], Macenko [42], and Vahadane [43] have been widely used to normalize stain colors by converting images to a standard color space and matching them to a reference. To this day, the Macenko [42] algorithm and the contemporary version of the Reinhard algorithm introduced in Roy et al. (2021) are the gold-standards for stain normalization. More recently, however, deep-learning methods have emerged as a powerful alternative. Models like StainNet [44], CycleGAN [45], and StainGAN [46] utilize neural networks to learn and translate between different stain styles. This approach offers increased robustness and a reduced tendency for the numerical instabilities that can sometimes affect traditional methods.

After the instances have been processed, the next step in many computational pathology workflows is feature extraction. This process converts raw image data into a numerical representation that can be used by machine learning models. The decision of whether to perform this step, and which method to use, depends on the downstream model.

Traditional machine learning models, such as Support Vector Machines (SVMs) and Random Forests, aren't designed to work directly with raw image pixels, so explicit feature extraction is essential [8]. In contrast, modern end-to-end deep learning models like

Convolutional Neural Networks (CNNs) and Vision Transformers (ViTs) automatically learn features during the training process, eliminating the need for a separate extraction step [8][48].

Other deep-learning architectures, such as Multiple Instance Learning (MIL), typically use a CNN or ViT as a feature extractor head prior to further computation. While pre-trained general-purpose models like ResNet50 [48] are commonly used, there has been a recent increase in extractors specifically trained on pathological features, such as CONCH [49] or Virchow [50]. Whether this extractor is "fixed" (i.e., a pre-trained, unlearning module) or fine-tuned depends on the specific model's configuration.

## The Problem and Proposed Solution

Currently, the largest obstacle in preprocessing for computational pathology is the lack of standardization , which decreases reproducibility [7][17][18]. Most studies often create their own preprocessing workflows and do not share them, which makes it difficult to recreate their results. Although frameworks like CLAM [15], STAMP [35] , SlideFlow [51], pathML [52], and TiaToolBox [53] provide preprocessing capabilities, they are often tied and optimized to their own end-to-end methodologies for patch to feature extraction. This design can limit their adaptability to research that requires more flexibility in pipeline design, ultimately, necessitating the creation of individualized pipelines.

For instance, CLAM [15], a highly regarded framework for WSI preprocessing, is primarily focused on tissue segmentation, patching, and feature extraction with their in-house extractors (e.g., UNI [54], CONCH [49]). CLAM [15] does not natively include stain normalization, comprehensive quality control, or resolution conversions. Similarly, STAMP's [35] and pathML's [52] primary focus on feature extraction restricts the flexibility needed for

more complex deep-learning tasks. This design makes it difficult for researchers to use their own personalized feature extractors or to bypass feature extraction entirely to use the preprocessed images as a direct input for applications such as training a convolutional neural network or vision transformer. Furthermore, it limits the application of commonly used techniques like image augmentation, which is a quintessential technique to help improve model performance and reduce overfitting to the training set [55][56].

**Table 1.1. Features available in SlideLab in comparison to other preprocessing pipelines. I**

*In the tiling "Tiling at level 0" refers to the fact that there is no consideration of resolution differences between samples.*

| | GPU Acceleration | Stain Normalization | Tiling | Quality Check | Error Report | Summary Report | Annotations (Import) |
|---|---|---|---|---|---|---|---|
| **SlideLab** | During stain normalization, laplacian variance, feature extraction | Macenko [42], Vahadane [43], Reinhard [47], StainNET [44] | Resolution conversions (e.g., 40x to 20x), support for mpp and magnification. | Pen marks, tissue fold, tissue %, out-of-focus sections, foreign objects, adipose tissue | .csv with location in pipeline, | .csv with performance and processing metrics, reconstructed slide | Qupath, ASAP, GrandQC [26], Binary Masks |
| **CLAM [15]** | Feature extraction | | Tiling at level 0 | Background | | Reconstructed slide, | |
| **STAMP [35]** | Feature extraction | | Resolution conversions (e.g., 40x to 20x), support for mpp and magnification. | Background | | | |
| **SlideFlow [51]** | Feature extraction | Macenko [42], Vahadane [43], Reinhard [41], CycleGAN [45] | Resolution conversions (e.g., 40x to 20x), support for MPP and magnification. | Background, out-of-focus, pen marks | | .pdf | Qupath, SlideFlow studio, DeepBlur |
| **PathMl [52]** | Feature extraction | Macenko [42], Vahadane [43], Reinhard [41], | Tiling at level 0 | | | | |
| **TiaToolBox [53]** | Feature extraction | Macenko [42], Vahadane [43], Reinhard [41], | Resolution conversions (e.g., 40x to 20x), support for MPP and magnification. | Background | | | |

Building upon these observations, we developed a new framework, SlideLab, to directly address these key challenges in flexibility and standardization. Unlike existing frameworks,

SlideLab is designed with a modular architecture, allowing researchers to construct highly customized preprocessing pipelines adjusted to their needs. It provides comprehensive modules for essential steps often missing in other tools, including advanced quality control and various methods for stain normalization (Table 1.1). The framework's support for annotation inputs allows users to define specific regions of interest and integrate the outputs of other tools (e.g., GrandQC [26]) to refine their preprocessing outputs. SlideLab is optimized to create and store high-quality tile datasets, which is its recommended application. However, it also offers the flexibility to proceed directly to feature extraction as an optional, integrated step. This design enables extreme flexibility for researchers while providing them with high-quality datasets for their downstream tasks. By prioritizing modularity and flexibility, SlideLab supports a wider range of research methodologies, ultimately improving the reproducibility and adaptability of computational pathology studies.

The following sections will detail the methodology, implementation and benchmarking of SlideLab against existing tools, demonstrating how it addresses quality control and computational performance in preprocessing as well as its effects in model performance.

## 1.2. SlideLab Pipeline

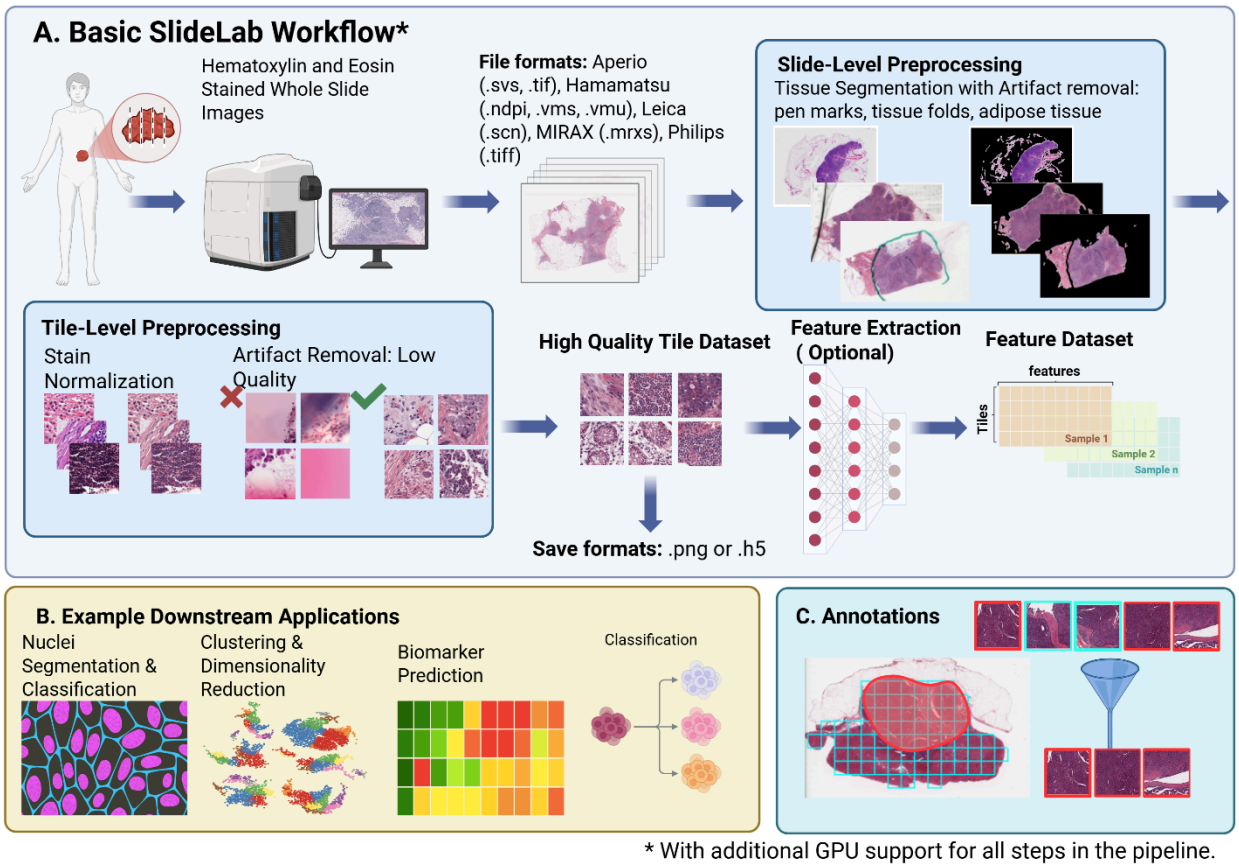


**Figure 1.2: The SlideLab workflow.**

*(A) The core workflow begins with whole slide images (WSIs), which undergo slide-level preprocessing for tissue segmentation and artifact removal. This is followed by tile-level preprocessing for tasks like stain normalization. The output is a high-quality tile dataset, with an optional step for feature extraction. (B) The resulting data can be used for various downstream applications, including nuclei segmentation and biomarker prediction. (C) The framework also supports annotations, allowing users to define specific regions of interest for analysis.*

### 1.2.1. Slide-Level Preprocessing: TissueMask Class

The TissueMask class in SlideLab is the initial, slide-level preprocessing step in the pipeline. It is responsible for identifying and isolating relevant tissue regions in a WSI while removing non-relevant sections (e.g., background, pen-marks, adipose tissue, tissue folds, cracked slides, foreign objects). Adipose tissue is purposefully excluded because, during slide preparation, its lipid content is removed. This leaves behind large, broken vacuoles with low cellularity that distort image statistics and are often incompatible with downstream methods such as stain normalization (**Figure 1.3**). Furthermore, adipose tissue content varies significantly on the preparation of the slides; even samples from the same tissue biopsy can exhibit large adipose tissue differences due to manual processing. Consequently, this can cause models to learn spurious associations. For example, in solid-tumor specific analysis, where the focus is on neoplastic cells and the tumor microenvironment, the presence of adipose tissue has been shown to significantly decrease model performance [27][40][57][58].

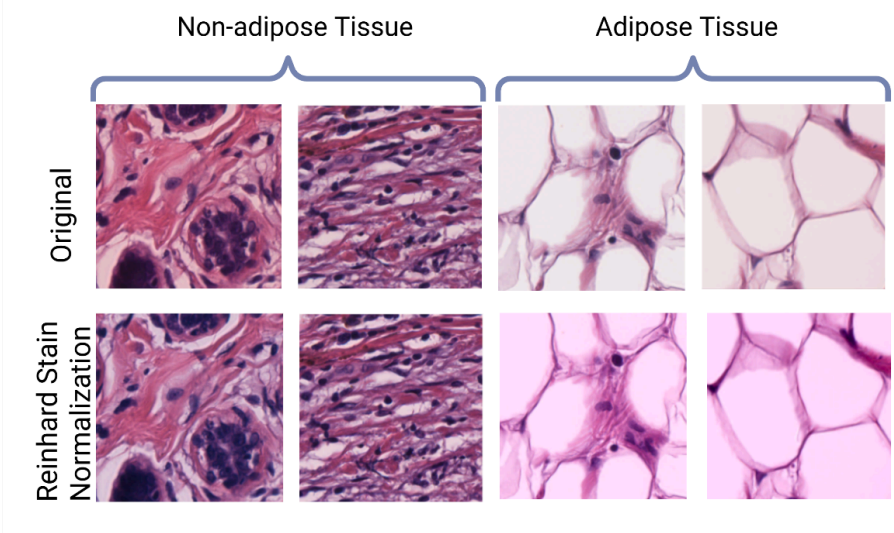


**Figure 1.3: Effects of adipose tissue versus non-adipose tissue on stain normalization algorithms like Reinhard [41].**

Once the TissueMask object has been created, it is used to calculate the candidate patches that pass through a specified tissue content threshold (the default is 70%). The following subsections detail the core algorithms used in the TissueMask class.

#### *1.2.1.1 Automatic Thresholding: Tissue Segmentation*

SlideLab distinguishes tissue from background and common artifacts, such as pen marks, using a method adapted from Schreiber et al. (2024). In H&E stains, tissue regions typically exhibit dominant red and blue channels, with low green intensities. This method, as described in *Algorithm 1.1*, takes advantage of this by first normalizing the RGB image (downscaled WSI) and then computing two intermediate representations of these color differences: $I_{R-G}$ (i.e., the red-green intensity difference; eosin-stained components) and $I_{B-G}$ (i.e., the blue-green intensity difference; hematoxylin-stained components).

In contrast to the original approach, which uses the element-wise product of $I_{R-G}$ and $I_{B-G}$, SlideLab constructs its Tissue Representation ($T$) using the element-wise minimum of the two. By requiring a strong intensity difference in both components, this approach penalizes regions with dominant singular-component intensities (red and blue pen marks) but favors areas where

both components are present, even if weaker (such as in lighter tissue) (**Figure 1.4**).

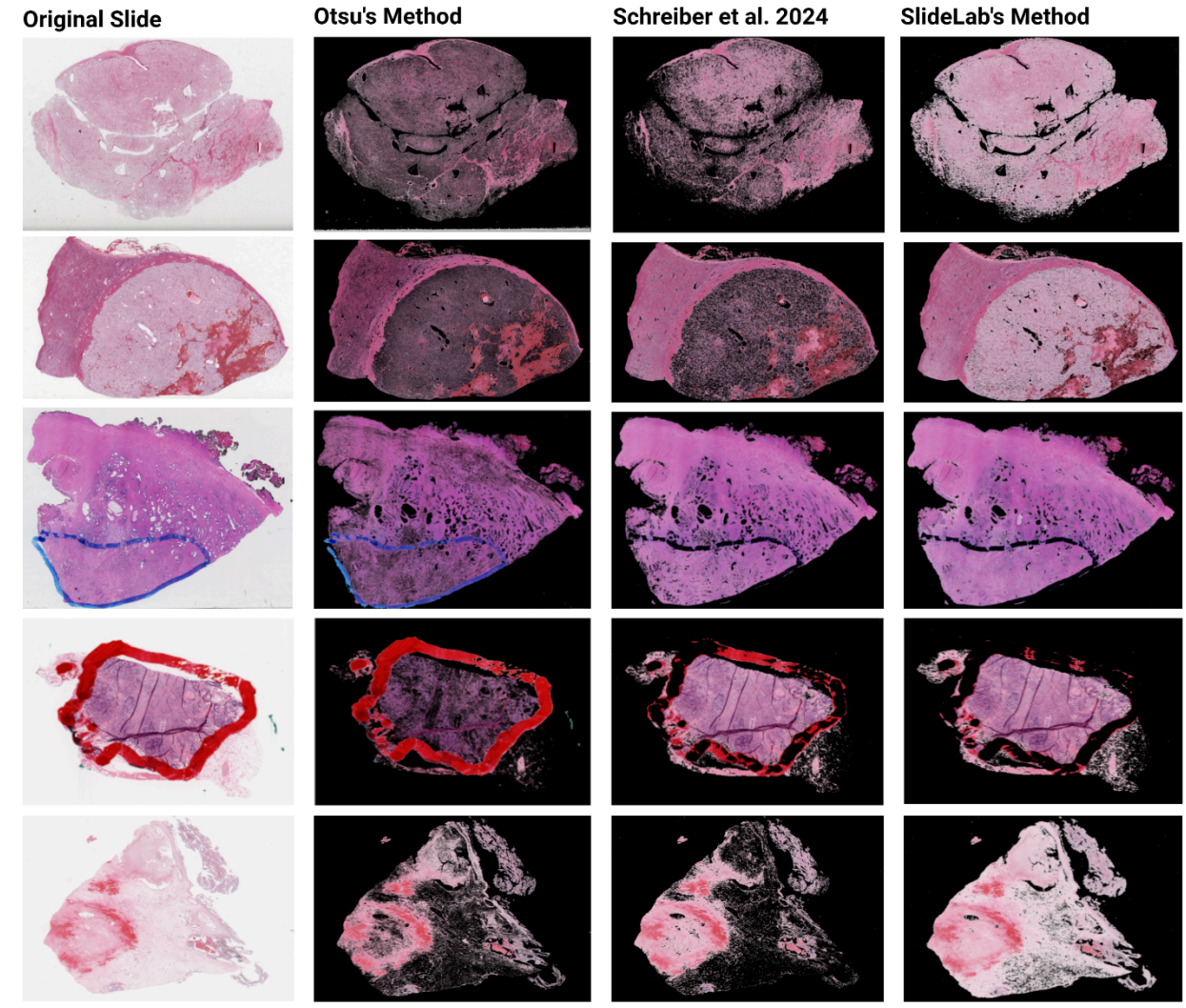


**Figure 1.4: Comparison of tissue thresholding using Otsu's method, Schreiber et al. 2024, and SlideLab's method.**

*The leftmost panel shows the original tissue slide, while the subsequent panels display the tissue detected by each method. In these panels, areas excluded by the "tissue detection" mask appear in black, whereas detected tissue retains its original colors.*

Finally, once *T* has been defined, Otsu's method [30] is applied to *T* to find a threshold that separates tissue from the background. Pixels that pass this threshold are considered tissue. This algorithm is the foundation of the SlideLab *TissueMask* class, where other complementary algorithms are utilized to remove other artifacts missing in the initial filtering.

**Algorithm 1** SlideLab's method for segmenting H&E stained tissue. Modified from Schreiber et. al. (2024).

**Input:** RGB Image: $[I_R, I_G, I_B]$
**Normalize:**
  $I_R \leftarrow I_R/255$
  $I_G \leftarrow I_G/255$
  $I_B \leftarrow I_B/255$
**R - G Representation:**
  $I_{R-G} \leftarrow \text{ReLU}[I_R - I_G]$
**B - G Representation:**
  $I_{B-G} \leftarrow \text{ReLU}[I_B - I_G]$
**Tissue Representation:**
  $T \leftarrow \min(I_{R-G}, I_{B-G})$
**Otsu threshold:**
  $\gamma \leftarrow \text{Otsu}[T]$
**for** each pixel $p$ in the image **do**
  **if** $T[p] > \gamma$ **then**
    Pixel $p$ is segmented as tissue
  **else**
    Pixel $p$ is rejected
  **end if**
**end for**
**Output:** Segmented tissue regions

**Algorithm 1.1. SlideLab's method for segmenting H&E-stained tissue.**
*This pseudocode outlines the steps for tissue segmentation using a modified version of the Schreiber et al. (2024) algorithm.*

#### *1.2.1.2 Artifact Removal: Tissue Folds*

To address the issue of tissue fold artifacts, SlideLab implements the tissue fold removal method as detailed in Kothari et. al. (2013). This algorithm detects tissue folds in downscaled WSIs by using an adaptive soft thresholding technique. After converting the image into the HSV (Hue, Saturation, Value) color space, a heatmap (*D)* is produced by the element-wise subtraction of the saturation and value channels. In D, tissue fold regions would appear brighter as they are typically brighter (higher value) and less colorful (lower saturation) than other tissue types. Two adaptive thresholds, soft and hard, for tissue folds, are then determined based on the object connectivity within *D*. Object connectivity is determined using the python library OpenCV, specifically with the method *connectedComponentsWithStats.* A pixel is identified as a fold if its value in *D* exceeds the soft threshold and it's near a pixel above the hard threshold. The full algorithm is detailed in *Algorithm 1.2*.

**Algorithm 2** Tissue Fold Detection Algorithm (Kothari et. al.)

```
function GetSaturationIntensity(Image_RGB)
    Image_HSV ← Convert Image_RGB to HSV color space
    Equalize histogram of the Value channel of Image_HSV
    S ← Saturation channel of Image_HSV (normalized)
    I ← Value channel of Image_HSV (normalized)
    return S, I
end function

function ComputeDifferenceImage(S, I)
    return S − I
end function

function GetConnectedObjectCounts(DifferenceImage, Thresholds)
    Counts ← empty list
    for each t in Thresholds do
        BinaryImage ← (DifferenceImage > t)      ▷ Create binary image
        NumLabels ← Find connected components in BinaryImage
        Counts.append(NumLabels − 1)             ▷ Exclude background label
    end for
    return Counts
end function

function FindThresholds(Counts, Thresholds, α, β)
    MaxCount ← Maximum value in Counts
    T(c) ← Largest threshold t ∈ Thresholds where Count at t ≥ c
    T_hard ← T(α · MaxCount)
    T_soft ← T(β · MaxCount)
    return T_hard, T_soft
end function

function ApplyThresholds(DifferenceImage, T_soft, T_hard)
    SoftMask ← (DifferenceImage > T_soft)
    HardMask ← (DifferenceImage > T_hard)
    Kernel ← 5x5 square kernel
    DilatedHardMask ← Dilate HardMask with Kernel
    CombinedMask ← Element-wise AND of SoftMask and DilatedHardMa
    return CombinedMask
end function

function DetectTissueFolds(InputImage, α = 0.65, β = 0.34, MinSize = 64)
    S, I ← GetSaturationIntensity(InputImage)
    DifferenceImage ← ComputeDifferenceImage(S, I)
    Thresholds ← Generate a range of values from -1.0 to 1.0
    Counts ← GetConnectedObjectCounts(DifferenceImage, Threshol
    T_hard, T_soft ← FindThresholds(Counts, Thresholds, α, β)
    FoldMask ← ApplyThresholds(DifferenceImage, T_soft, T_hard)
    FoldMask ← Remove small holes from FoldMask with MinSize
    return Invert FoldMask
end function
```

**Algorithm 1.2.** ***SlideLab's method for identifying tissue folds.***

*This pseudocode outlines the steps for tissue fold segmentation using the implementation from Kothari et. al. (2013).*

### *1.2.1.3 Artifact Removal: Pen Filtering*

While most artifacts are removed during the automatic thresholding step, residual artifacts are removed through a series of targeted filtering methods (Figure 1.5). These include specialized filters for blue, red, green, and black pen marks, as well as gray background removal. To optimize this process, these filters are only applied to pixels included after the automatic thresholding.

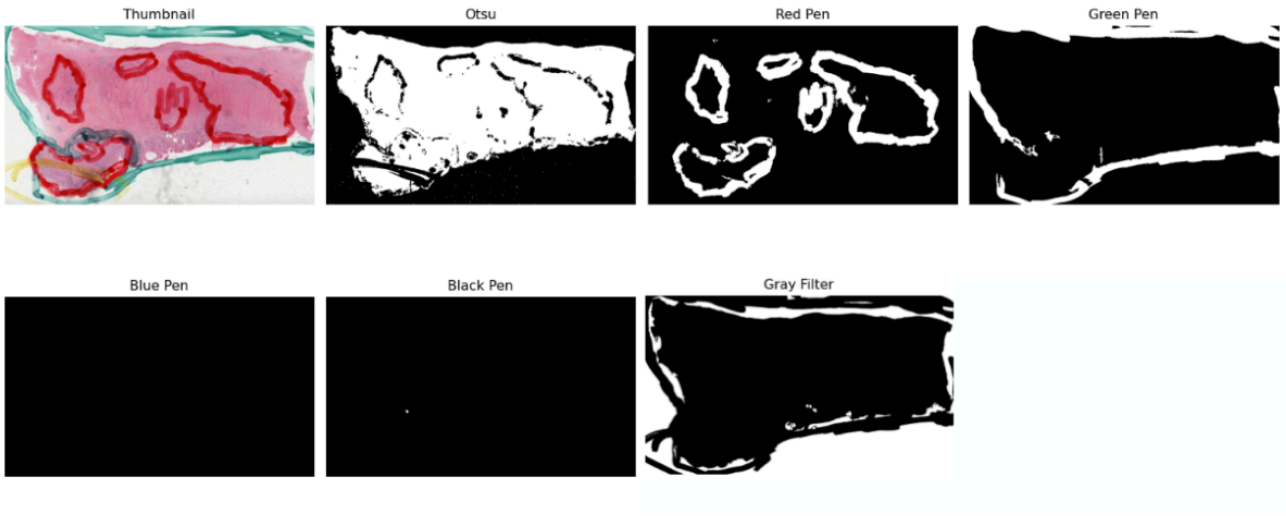


**Figure 1.5:** ***Example of pen filtering steps for a sample with excessive pen marks.***

Each of these methods operates on a downscaled version of the WSI and separates the image into its RGB (red, green, blue) channels. Color-based thresholding is then used to isolate artifact regions, with each method having a set of threshold triplets (e.g., red: 60, blue: 20, green: 50) to progressively remove pixels within fixed ranges of intensity [36]. For example, the blue pen filter checks for pixels with low red and green intensity but high blue intensity. Logical masks are created per channel and are combined to create a binary mask of potential blue ink

regions. To ensure full removal, a morphological dilation step expands the mask to cover marked areas.

Additionally, because color-based thresholds may occasionally capture true tissue due to staining variability in H&E slides, a safeguard is implemented to limit the area of the filtered mask. Specifically, each mask is constrained to cover no more than a user-defined percentage (e.g., 30%) of the initially detected tissue area. If this limit is exceeded, the method assumes the filter is over-filtering and returns an empty mask instead.

#### *1.2.1.4 Artifact Removal: Small Object & Hole Removal*

Once all masks have been generated and combined, some tiny holes or specks may remain because of processing noise. These are removed by identifying and removing connected components in the binary mask that fall below a minimum area threshold, under the assumption that they represent noise rather than true tissue.

This is implemented using OpenCV's *connectedComponentsWithStats'* function, which identifies each connected region and calculates its area. Components smaller than the allowed minimum size are removed, and holes within larger regions are filled. To avoid accidentally removing too much tissue or introducing background, the method monitors the percentage of the mask that remains after filtering. If the retained tissue drops below a user-defined threshold (e.g., 90%), the minimum size is iteratively reduced, allowing smaller components to remain.

**Algorithm 3** Small object and Hole Removal

**Input:** $M$: Binary mask
$min\_size_{init}$: Minimum area threshold (default: $0.0001 \times M$.size)
$avoid\_overmask$: Boolean
$overmask\_thresh$: Percentage threshold
$kernel\_size$: Dilation kernel dimension
**Output:** $M_{cleaned}$: Cleaned binary mask

**Data:** $min\_size \leftarrow min\_size_{init}$ (use default if $min\_size_{init}$ is None)
$M \leftarrow$ Convert to UINT8 and Threshold($M$)

$(num\_labels, labels, stats) \leftarrow$ ConnectedComponentsWithStats($M$)
$areas \leftarrow stats[1:, \text{CC\_STAT\_AREA}]$

**while** *True* **do**
  $M_{cleaned} \leftarrow$ ZerosLike($M$)
  **for** $i \leftarrow 1$ **to** $num\_labels - 1$ **do**
    **if** $areas[i-1] \geq min\_size$ **then**
      $M_{cleaned}[labels == i] \leftarrow 255$
    **end**
  **end**
  $mask\_percentage \leftarrow (\sum(M_{cleaned} > 0)/M_{cleaned}.\text{size}) \times 100$
  **if** ***not*** $avoid\_overmask$ ***or*** $mask\_percentage < overmask\_thresh$ ***or*** $min\_size < 1$ **then**
    **break** ▷ Exit loop if overmasking prevention not needed or threshold met
  **end**
  $min\_size \leftarrow \lfloor min\_size/2 \rfloor$ ▷ Reduce min_size for re-filtering
**end**

**if** $kernel\_size > 1$ **then**
  $M_{cleaned} \leftarrow$ Dilate($M_{cleaned}$, kernel_size)
**end**
**return** $M_{cleaned}$

**Algorithm 1.3. SlideLab's method for small object and hole removal.**

*This pseudocode outlines a method for refining a binary mask by iteratively removing small objects and filling holes. The algorithm takes an input binary mask and a minimum area threshold, then processes the image to remove components smaller than the threshold, returning a cleaned mask. The process includes a loop for iterative refinement and a final dilation step to adjust the kernel size.*

### 1.2.2 Tile-Level Preprocessing: SlidePreprocessing Class

The *SlidePreprocessing* class in SlideLab handles all tile-level processing after a *TissueMask* object is created. Its main task includes normalizing H&E-stain intensities, adjusting for differences in WSI resolutions, dividing WSIs into tiles, and assessing whether tiles are in focus. As one of the most elaborate portions of the pipeline, it is designed for high performance and runs in parallel using a producer-consumer queue pattern for synchronization. To ensure fast computation, all processing operations—other than I/O-based steps—are implemented using optimized libraries such as Numba, OpenCV, and Pytorch. The Pytorch library implementations additionally allow for GPU capabilities. The following subsections describe the associated steps and mechanisms in the *SlidePreprocessing* class.

#### *1.2.2.1 Resolution Conversions & Sanity Checks*

As shown in Figure 1.1A, resolution conversions are necessary to extract patches of the same physical size in microns across slides scanned at different magnifications. To achieve this, the user is asked to provide the desired resolution either as microns-per-pixel (MPP) or objective lens power (i.e., native magnification) and the desired patch size in pixels. SlideLab uses Openslide to extract metadata such as MPP and the objective lens power (e.g., 0.5 MPP or 20x) for each slide. The conversion is summarized in Equation 1.1.

$$\text{Patch Size}_{\text{orig}} = \text{Patch Size}_{\text{desired}} \times \left( \frac{\text{MPP}_{\text{desired}}}{\text{MPP}_{\text{orig}}} \quad \text{or} \quad \frac{\text{Mag}_{\text{orig}}}{\text{Mag}_{\text{desired}}} \right) \tag{1}$$

**Equation 1.1. Patch size**

Additionally, SlideLab enforces a constraint against upsampling: patches cannot be extracted at a higher resolution than the slide's native scan resolution. If the requested resolution exceeds the base resolution for a slide, an error is raised to prevent interpolation artifacts and that slide is skipped.

#### *1.2.2.2 Candidate Patch Selection*

To determine valid patch sampling locations within a WSI, a grid of coordinates is generated based on the adjusted patch size—obtained from Equation 1.1 —and the provided overlap per patch. The stride, which represents the number of pixels between the top-left corners of consecutive patches, is calculated as:

$$\text{stride} = \begin{cases} \frac{size}{overlap} & \text{if } overlap > 1 \\ size & \text{otherwise} \end{cases} \tag{2}$$

**Equation 1.2. Patch stride**

Using this stride, patch coordinates are calculated and mapped to the *TissueMask* object. Each candidate is checked against the *TissueMask* to quantify the tissue content of the corresponding patch in the mask. Candidate coordinates whose masked patch exceeds the tissue percentage threshold are kept for WSI patching.

#### *1.2.2.3 Producer-Consumer Framework and Computational Resource Management*

SlideLab uses a producer-consumer queue system to parallelize tile processing and saving. A *TileIterator* object is initialized, where each index corresponds to a candidate patch coordinate and its associated tile. These indices are added to an Index Queue, which is consumed by producer workers that load, process, and filter each tile. Valid tiles and their coordinates are then placed into a Save Queue, where consumer workers retrieve and save the tiles concurrently.

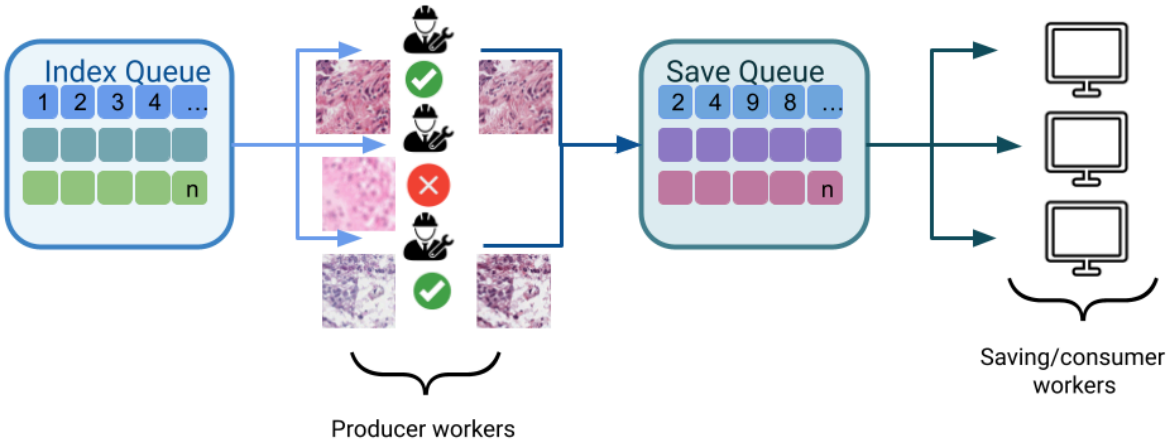


**Figure 1.6:** ***Producer-Consumer Framework and Computational Resource Management***

The output format and available device (CPU or GPU) determine how resources are allocated for tile processing and saving. When a GPU is available, indices are grouped into batches rather than passed individually to leverage parallel GPU computation. In this case, a *TileDataloader* with multiple workers is used to load batches efficiently, while a larger number of consumer workers handle saving to balance GPU throughput with I/O speed. Additionally, if the *.h5* format is selected, only a single dedicated worker performs batched saving of tiles and associated metadata in intermittent flushes while all other available workers are set as producers.

On the other hand, selecting the *.png* format triggers multiple threads across several saving workers to save tiles in parallel more efficiently.

If no output format is selected, a *TileProcessorDataloader* is created to directly create batches of pre-processed tiles for direct feature extraction.

#### *1.2.2.4 Artifact Removal: Blurry Tile Removal*

SlideLab implements a Laplacian filter to detect out-of-focus tiles. In this method, an input RGB tile is converted to grayscale using standard luminance weights. Then, a Laplacian kernel is convolved with the grayscale tile to highlight areas of rapid intensity change (i.e., edges and fine details). The variance of the resulting Laplacian image quantifies the sharpness and focus of the original tile. For example, a low variance means that there is a lack of defined edges, and thus the tile is considered blurry. SlideLab uses a threshold-based method to determine whether a tile is blurry or not. As a default, tiles with a variance less than 0.025 are considered blurry, however, the user can redefine this threshold. This algorithm is implemented in both Torch and OpenCV to allow for GPU and CPU speedups.

**Algorithm 4** Laplacian Blur Detection

**Input:** $Img$: Input image tensor (e.g., $(C, H, W)$ or $(H, W, C)$)
$Var_{threshold}$: Variance threshold (float)
**Output:** $IsBlurry$: Boolean
$Variance$: Float

**if** $Img.dim() == 3$ ***and*** $Img.shape[0] == 3$ **then**
  $Img_{gray} \leftarrow 0.2989 \times Img[0] + 0.5870 \times Img[1] + 0.1140 \times Img[2]$
**end**
**else if** $Img.dim() == 3$ ***and*** $Img.shape[-1] == 3$ **then**
  $Img_{gray} \leftarrow 0.2989 \times Img[\ldots, 0] + 0.5870 \times Img[\ldots, 1] + 0.1140 \times Img[\ldots, 2]$
**end**
**else**
  **Raise Error:** "Invalid image dimensions"
**end**
$Img_{gray} \leftarrow \text{Reshape}(Img_{gray}, (1, 1, H, W))$
$Kernel_{Laplace} \leftarrow \text{Tensor}([[0, 1, 0], [1, -4, 1], [0, 1, 0]])$
$Img_{Laplace} \leftarrow \text{Conv2D}(Img_{gray}, \text{Reshape}(Kernel_{Laplace}, (1, 1, 3, 3)), \text{padding} = 1)$
$Variance \leftarrow \text{Variance}(Img_{Laplace}).\text{item}()$
$IsBlurry \leftarrow (Variance \leq Var_{threshold})$
**return** $IsBlurry, Variance$

**Algorithm 4. Laplacian Blur Detection**
*This pseudocode outlines a method for calculating the Laplacian variance of an image. The algorithm takes the input of a single or batch of tiles. The input images are transformed into grayscale using standard luminance weights. The grayscale images are then passed and convolved through a Laplacian kernel. The Laplacian variance is obtained from the convolved result.*

#### *1.2.2.5 H&E Normalization*

SlideLab provides a comprehensive suite of stain normalization methods, including the traditional Macenko [42], Reinhard [47], and Vahadane [43] algorithms, as well as the deep-learning-based StainNet model [44].

To enhance computational efficiency, all traditional methods are accelerated using the Numba library, which compiles Python code into machine code, yielding a 5x to 10x improvement in throughput. Additionally, these algorithms are implemented in Torch for GPU speedups using a modified version of TorchStain's[59] methodology, which enables batching to further optimize performance and GPU utilization.

### 1.2.3 Annotations: AnnotationReader Class

SlideLab allows for the usage of previously annotated sections of the slides to be utilized as TissueMasks to limit preprocessing to specific regions of interest. The *AnnotationReader* class takes input files like *.tiff, .npy,* and, *geojson* to create a binary tissue mask. The user can define if they wish this region to be further processed (e.g., remove adipose tissue) by the *TissueMask* class or not. As SlideLab is mainly based on heuristic algorithms, this allows users to upload annotations from more complex artifact-filtering algorithms like GrandQC [26] or HistoQC [27] on a need-to basis. Additionally, as some datasets may include the pre-defined tumor regions, this allows for direct extraction of these regions.

### 1.2.4 Feature Extraction: SlideEncoding Class

The *SlideEncoding* class in SlideLab is responsible for providing optional capabilities for feature extraction. The class initiates a pre-trained feature extractor like Resnet50 [48], applies the necessary transformations to the preprocessed tiles, and then obtains the feature vectors as an output. The feature vectors, coordinates, and other metadata that might be related to the tile are stored in a *h5* file in batches asynchronously to prevent I/O hangups. Currently SlideLab supports the following pre-trained feature extractors: Resnet50 [48], CONCH [49], UNI [54], STAR [64], DinoBloom [63], and Virchow [50]. To allow access to restricted pre-trained model weights, SlideLab allows connection through generated access tokens from *HuggingFace*. This optional step allows researchers to either work directly with the high-quality image tiles or to generate a more compact feature representation, depending on their downstream analysis goals.

### 1.2.5 Preprocessing Summaries: Report Class

SlideLab's *Report* class provides a comprehensive overview of the preprocessing pipeline's quality and performance. This class generates three main types of reports: a summary report, which logs wall time, CPU time, and key tile metrics for each step; an error report, which tracks and logs any failures that occur, including the specific sample and step; and a quality report, which provides visual and statistical outputs. The quality report is particularly useful for validation, as it can show side-by-side examples of a random tile before and after stain normalization, provide visual examples of tiles removed due to blurriness, and display a graph of the Laplacian variance distribution to illustrate the effect of quality control thresholds. Furthermore, if the “reconstruct slide” option is toggled, it will create a stitch image of the tiles mapped to their spatial coordinates.

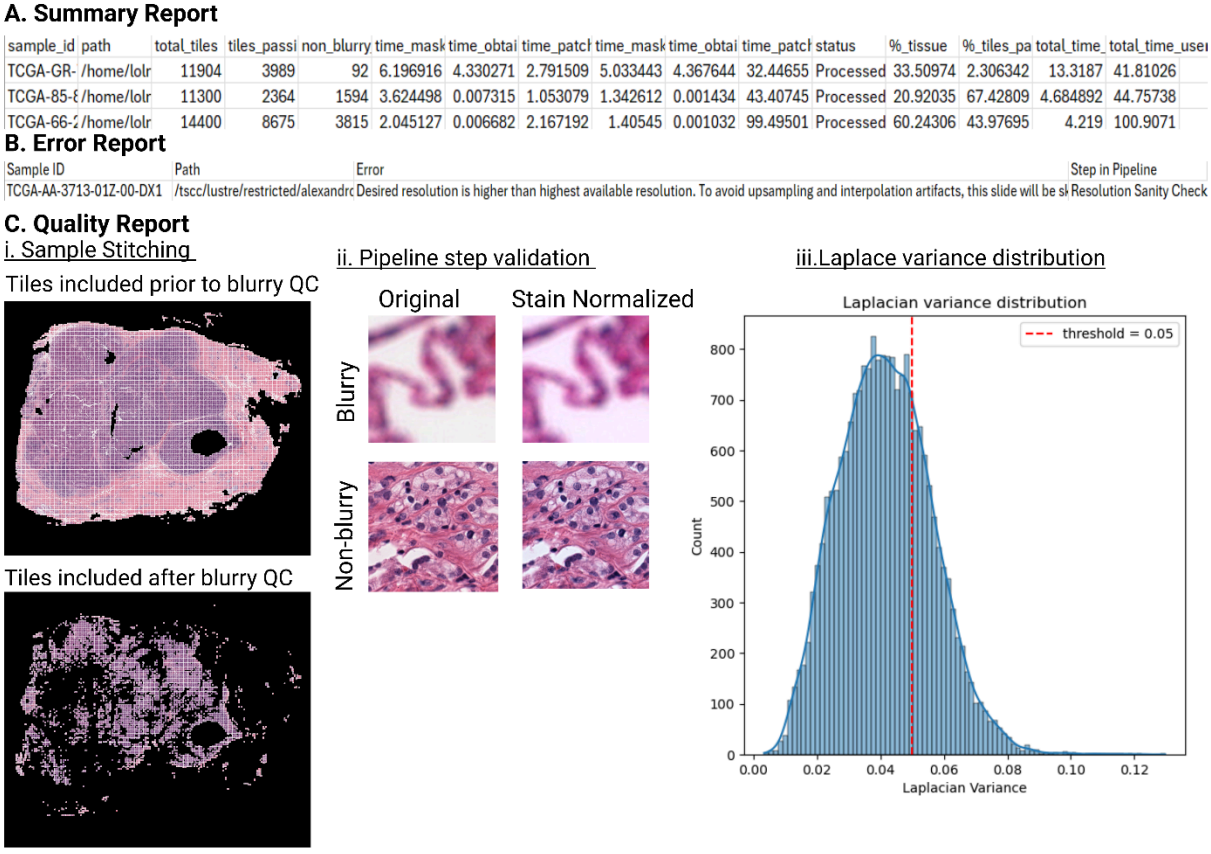


**Figure 1.7: Preprocessing Summaries Report**

*(A) The summary report, detailing sample performance statistics. (B) The error report, identifying failed samples and their point of failure. (C) The quality report, visualizing the effects of tile filtering. This includes the Laplacian variance distribution to guide threshold selection, alongside examples of removed and normalized tiles.*

## 1.3 Testing Methodology

To evaluate pipeline performance, SlideLab was benchmarked against CLAM [15] , SlideFlow [51] , pathML [52], TiaToolBox [53] , and STAMP [35] under standardized conditions. The latest versions available on their respective GitHubs were used. All tools were configured with default parameters as specified in their respective documentation to ensure a fair comparison. For all experiments, tiles were extracted at a 20x objective lens (0.5 MPP) with a target size of 256 pixels and no overlap between the tiles.

The experiments were performed on uniform hardware: Intel Xeon Platinum 16-core CPU processors with a standard 3.5 GB memory per core. When applicable, a NVIDIA A100 GPU was used. The following subsections detail the methodology for performance assessment.

### 1.3.1 Computational Performance

To evaluate computational performance, we benchmarked each tool across a range of increasing sample sizes (1, 10, 50, 100, and 1,000 whole slide images). Each sample size was tested with 10 trials. To ensure a consistent computational load for each trial, we processed a randomly preselected subset of WSIs from The Cancer Genome Atlas (TCGA) Colon Adenocarcinoma (COAD) cohort. For each trial, we recorded wall time, system time, and memory usage. Time measurements were collected using Python's time module. Memory consumption was sampled every 0.05 seconds using *psutil*, and the median memory usage from these samples was retained for analysis.

All pipelines, with the exception of SlideLab, were executed using their default stepwise configurations as specified in their documentation. This typically involved simple tiling without any normalization. Consequently, SlideLab was benchmarked in two configurations: its tiling

abilities alone (serving as a baseline) and its performance with additional tile-level processing steps, such as tile quality control and stain normalization.

Since STAMP [35] only allows for direct feature extraction without an intermediate saving step, it was configured to use its "empty" encoder option, which simply returns the coordinates in microns of valid tiles.

### 1.3.2 Quality Control Performance

To evaluate quality control performance, all tools were benchmarked against GrandQC, the current gold standard for WSI quality assessment with a dice score of 0.938 on segmentation of tissue without artifacts [26]. Given the lack of publicly available, annotated WSI datasets specifically curated for artifact detection, GrandQC [26] is currently the most viable resource for benchmarking artifact detection and quality control in WSI preprocessing.

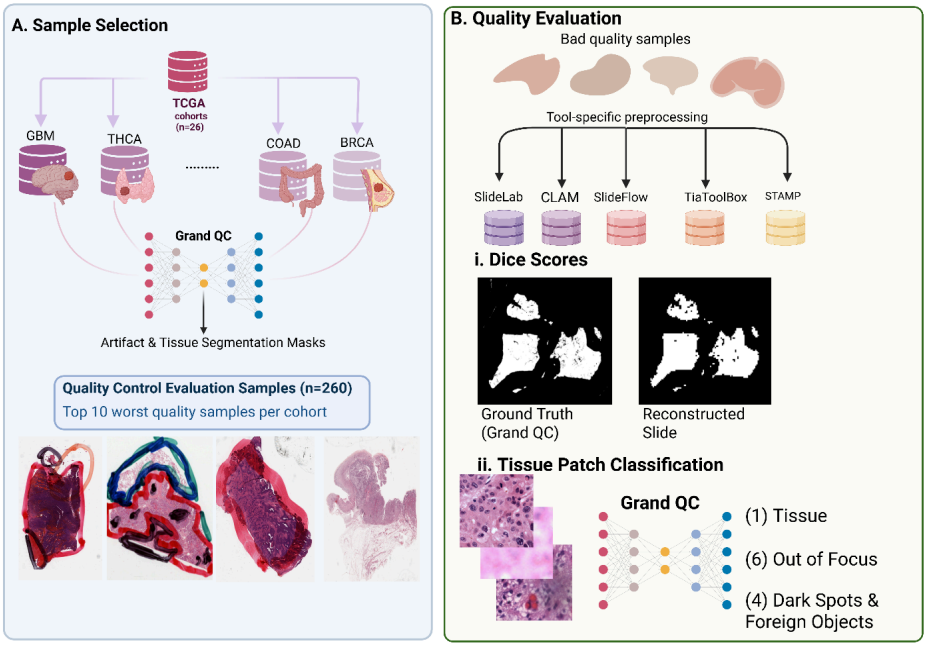


**Figure 1.8: Quality control evaluation benchmarking process**

*This figure shows the steps taken to evaluate the quality control abilities of the different benchmarkable tools. (A) Shows how the bad quality WSI cohort was selected. (B) Shows how the results of each tool were benchmarked.*

Using GrandQC's [26] multi-class masks spanning 26 TCGA cohorts, we identified the 10 most artifact-heavy samples per cohort based on non-background artifact prevalence, ensuring balanced representation across tissue types (Fig 1.8A). The selected whole-slide images (n = 260) were processed through each pipeline. Performance was assessed through two primary metrics: (I) the overall Dice score (Equation 1.3) comparing each tool's reconstructed slide (mapping valid tiles to their spatial location) to a binary presentation of GrandQC's [26] reference mask (i.e., tissue vs non-tissue), and (II) tile-level classification where GrandQC [26] was used to classify each generated tile into seven categories (segmented tissue, background, tissue folds, dark spots/foreign objects, pen markings, air bubbles/slide edge artifacts, and out-of-focus artifacts) as referenced in *Figure 1.2C*. For the tile-level classification, performance was measured as overall performance across the entire cohort (i.e., all tiles) and mean classification distribution at the sample-level.

$$\text{Dice Score} = \frac{2 \times \text{TP}}{(2 \times \text{TP}) + \text{FP} + \text{FN}} \tag{3}$$

**Equation 1.3. Dice Score**

These methods provide a quantitative measurement of overall true tissue detection and detailed artifact composition within the final dataset.

### 1.3.3 Model Performance

To isolate the impact of each preprocessing strategy on downstream model performance, a parallel set of experiments was conducted. In each experiment, a dedicated model was trained and evaluated exclusively on data generated by a single preprocessing tool. To achieve this, we implemented a standard attention-based Multi-Instance Learning (abMIL) framework with a pretrained ImageNet ResNet50 [48] as the feature extractor. The weights for the feature extractor were frozen during training. This structure was kept intentionally simple to isolate preprocessing effects from model architecture complexities.

#### *1.3.3.1 Classification Task*

The classification task was Microsatellite Instable (MSI) versus Microsatellite Stable (MSS) classification in colorectal cancer. Colorectal cancer is the second most common cause of cancer deaths in the United States. In 2025 alone, 154,270 new cases of colorectal cancer were diagnosed [60]. From these cases, approximately 15-20% express high-level microsatellite instability (MSI-H), which arises from a failure of mismatch repair genes (e.g., MLH1, MSH6) [61]. Due to this mismatch repair deficiency, MSI-H tumors tend to have a high mutational burden. This hypermutability leads to an increase of neoantigen production on the surface of MSI-H cancer cells, making them more vulnerable for detection by the immune system [62].

Because of this, the detection of MSI-H versus of MSS tumors has prognostic and therapeutic significance. On the account of the innate active immune system response in MSI-H cells, tumors with this biomarker are more likely to respond positively to immune therapy in comparison to chemotherapy [62]. On the other hand, MSS tumors are often described as “immune deserts” and benefit from chemo and radiation treatments [61].

#### *1.3.3.2 Datasets*

Two separate datasets were utilized for these tasks: (I) Mutographs, an internal dataset, and (II) TCGA-COAD. Each of these datasets were then separated into Flash-frozen (FF) and Formalin-Fixed Paraffin-Embedded (FFPE) to train two separate models.

In the FF set, the Mutographs dataset contained 422 MSI-H and 2076 MSS samples. The TCGA-COAD cohort contained 119 MSI-H and 415 MSS tumors.

In the FFPE set, the Mutographs dataset contained 103 MSI-H and 598 MSS samples. The TCGA-COAD cohort contained 60 MSI-H and 216 MSS tumors.

These were separated as each tissue preservation method greatly affects tissue morphology. As depicted in Figure 1.2C, FF slides are prone to freezing artifacts. Because of this, FF slides are often regarded as being "poorer" in quality than FFPE slides, specially for computational pathology slides. Testing both of these datasets allows us to better understand the effects of each preprocessing framework when dealing with both poor-quality and high-quality data for model training.

#### 1.3.3.3 Preprocessing

Each of the four total datasets (2 FFPE, 2 FF) were preprocessed individually by each tool being benchmarked. Tiles were extracted at a 20x (0.5MPP) resolution at 256 by 256 pixels with no overlap between tiles. All samples producing fewer than 150 tiles after tool-specific preprocessing were excluded as part of quality control.

To quantify how SlideLab's innate quality control or stain normalization affected model performance, we evaluated SlideLab's performance with only quality control (i.e., artifact removal) and with quality control and Macenko [42] stain normalization.

#### *1.3.3.4 Training*

The same abMIL structure was trained and evaluated in a 10-fold cross validation (CV) split per preprocessing tool. This meant that for each preprocessing tool two 10-fold CV schemes were implemented, one for the FF sets and another for the FFPE sets. To prevent data leakage, the 10-fold split was implemented at the patient-level rather than sample level. This stratification guaranteed that all slides from a single patient were exclusively contained within either the training or validation fold for each split to prevent patient-based bias.

For each preprocessing tool, the model was trained and validated on the Mutographs dataset. To assess the model's ability to generalize, the best performing weights from each fold were then evaluated on the completely independent, unseen TCGA-COAD test set.

In order to ensure reproducibility, all experiments were conducted with a fixed random seed (42) for all libraries (Pytorch, Python's random, Numpy, Sklearn) . An Adam optimizer with a learning rate of $1 \times 10^{-5}$ and a weight decay of $1 \times 10^{-6}$ was used. To address the large class imbalance, a weighted random sampler was used during epoch batch selection. Furthermore, the loss function—Cross Entropy Loss—was class-weighted, with the weights inversely proportional to the class frequencies of that fold's training set. No data augmentation was used. Each model was trained for 200 epochs with a patience of 20 epochs. The best weights per fold were saved by improvement in validation loss.

#### *1.3.4.5 Evaluation Metrics*

For each fold, the accuracy, precision, recall, F1, and Area Under the Curve (AUC) scores were recorded. The final evaluation contained the metrics from both the validation for each fold and the independent testing set. A Wilcoxon signed-rank test  was used to measure significance between the performance distributions across folds.

## 1.4 Results

### 1.4.1 Computational Performance

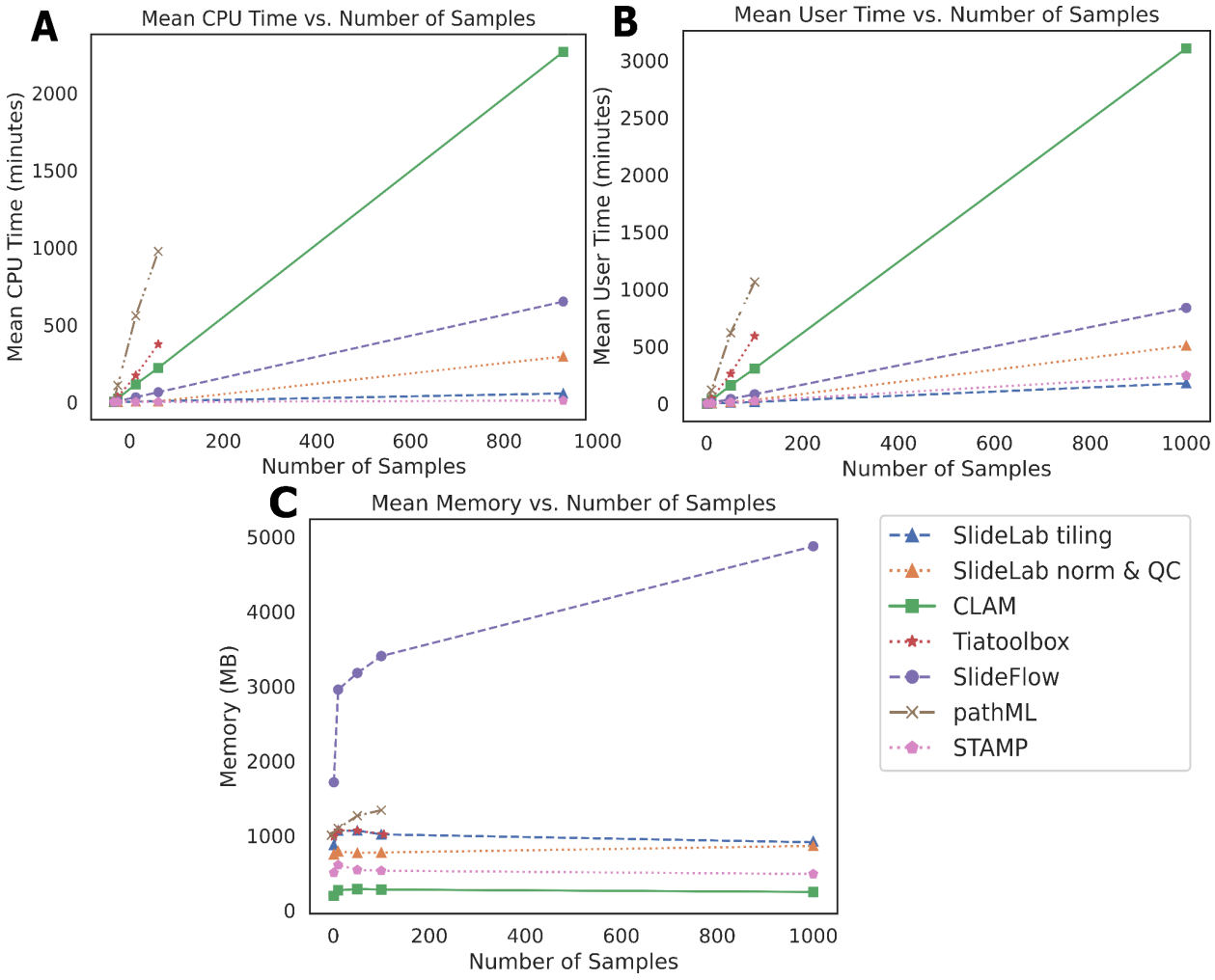

**Figure 1.9:** ***Computational performance benchmark of state-of-the-art WSI preprocessing pipelines against SlideLab.***

*The legend shows the different pipelines used. "SlideLab tiling" refers to SlideLab in its only tiling configuration (baseline). "SlideLab norm & QC" refers to using the full SlideLab pipeline, with stain normalization and quality control steps. In each plot, the average computational performance across ten trials for five different sample sizes is shown. (A) Mean CPU time (minutes). (B) Mean User time (minutes). (C) Mean Memory consumption (MB). TiaToolBox [53] and pathML [52] were unable to complete the 1,000 sample benchmarking.*

#### *1.4.1.1 Processing Time*

A near-linear relationship between mean processing time (CPU and user) and sample size was observed for all tools (Figure 1.9A, 1.9B). When focusing on the baseline tiling task at the 1,000 WSI scale, STAMP and SlideLab's baseline tiling configuration were the most efficient. While STAMP was fastest in pure CPU time (9.86 minutes vs. SlideLab's 55.86 minutes; 5.76x slower), its benchmark was configured to only extract tile coordinates without saving the image files. In contrast, even with the additional workload of writing tiles to disk, SlideLab was on average 1.37x faster in user time, completing the task in an average of 177.54 minutes compared to STAMP's [35] 244.95 minutes.

When comparing more similar pipelines, the performance gap widened significantly. CLAM [15] was the most computationally demanding tool, averaging 2,267.44 minutes of CPU time for the same tasks, with the SlideLab baseline being 40.59x faster in CPU time and 17.48x faster in user time. Similarly, SlideFlow [51] was on average 11.64x slower in CPU time and 4.71x slower in user time than SlideLab's baseline configuration.

Notably, SlideLab remained highly efficient even when running its full pipeline, which includes additional stain normalization and quality control steps. For instance, SlideLab's full pipeline at 1,000 WSIs was on average faster in CPU time than CLAM [15] (7.68x) and SlideFlow [51] (2.2x). Furthermore, it had a greater average speed in user time than CLAM [15] (6.13x) and SlideFlow [51] (1.65x). When compared to STAMP's [35] simpler coordinate extraction task, SlideLab's full pipeline remained competitive, running only 1.11x slower in user time, but in a similar trend to the baseline, running 12.31x slower in CPU time.

This trend was evident even at smaller scales, for instance, at the 100-WSI mark, SlideLab's full pipeline was already significantly more efficient than other tools. In CPU time,

SlideLab ran on average faster than CLAM [15] (41.89x) , SlideFlow [51] (12.12x), TiaToolBox [53] (71.1x), and pathML [52](184.8x). This same tendency was seen in user time (Fig 1.9B).

The scalability issues prevented TiaToolBox [53] and pathML [52] from completing the full benchmark as they encountered memory errors and unfeasible processing times past the 100 and 300 sample marks, respectively.

#### *1.4.1.2 Memory Consumption*

In comparison to other tools, both SlideLab configurations had relatively low memory consumption. The complete SlideLab pipeline reached a mean peak of just 850.76 MB, which is approximately 5.73x less than SlideFlow [51] , 5.87x less than pathML [52], and 1.25x less than TiaToolBox [53] at their respective mean peaks (Figure 1.9C). Both STAMP [35] and CLAM [15] had lower memory consumption than SlideLab, with mean peaks of 606.44 MB and 284.78 MB.

Overall, the SlideLab pipeline demonstrates a balance of speed and resource efficiency for practical WSI processing. It consistently outperformed most competing tools in both processing time and memory consumption, particularly at larger scales. While STAMP [35] recorded faster CPU times and lower memory usage, it is critical to note that this performance was achieved on a fundamentally simpler task that only extracted tile coordinates without saving the image files or further processing them. When considering the complete, end-to-end workflow required for analysis, SlideLab's performance proves it is a more robust and scalable solution while offering an expanded functionality.

## 1.4.2 Quality Control Performance

Due to issues with processing larger cohorts, as mentioned in section 1.4.1, pathML [52] was excluded from this evaluation. The following quality assessments rely on reference masks and tile-level classifications generated by the GrandQC [26] deep learning model, which as mentioned in the methodology, represents the current gold standard of automated artifact detection. While this provides a robust and scalable method for evaluation, it is important to note that these automations are subject to a minimal degree of error inherent to any deep learning based system.

### *1.4.2.1 Tissue Segmentation Accuracy*

The accuracy of each tool's tissue segmentation was quantified by comparing the reconstructed tissue mask (mapping the tiles to their spatial location in the WSI) against the GrandQC [26] reference using the Dice Similarity coefficient (Figure 1.10A). SlideLab achieved the highest mean 0.741 ± 0.015 Dice score, indicating that it was the framework with the most accurate and consistent tissue detection across the cohort. STAMP [35] performed similarly with a mean score of 0.714 ± 0.015, while CLAM [15] and SlideFlow [51] followed with scores of 0.665± 0.016 and 0.613 ± 0.019, respectively. TiaToolBox [53] had a significantly lower performance, with a mean Dice score of just 0.353 ± 0.012.

This trend was also shown in the medians with SlideLab's score of 0.815 outperforming STAMP [35] (0.779), CLAM [15] (0.719), SlideFlow [51] (0.707), and TiaToolBox [53] (0.359) (Figure 1.10A).

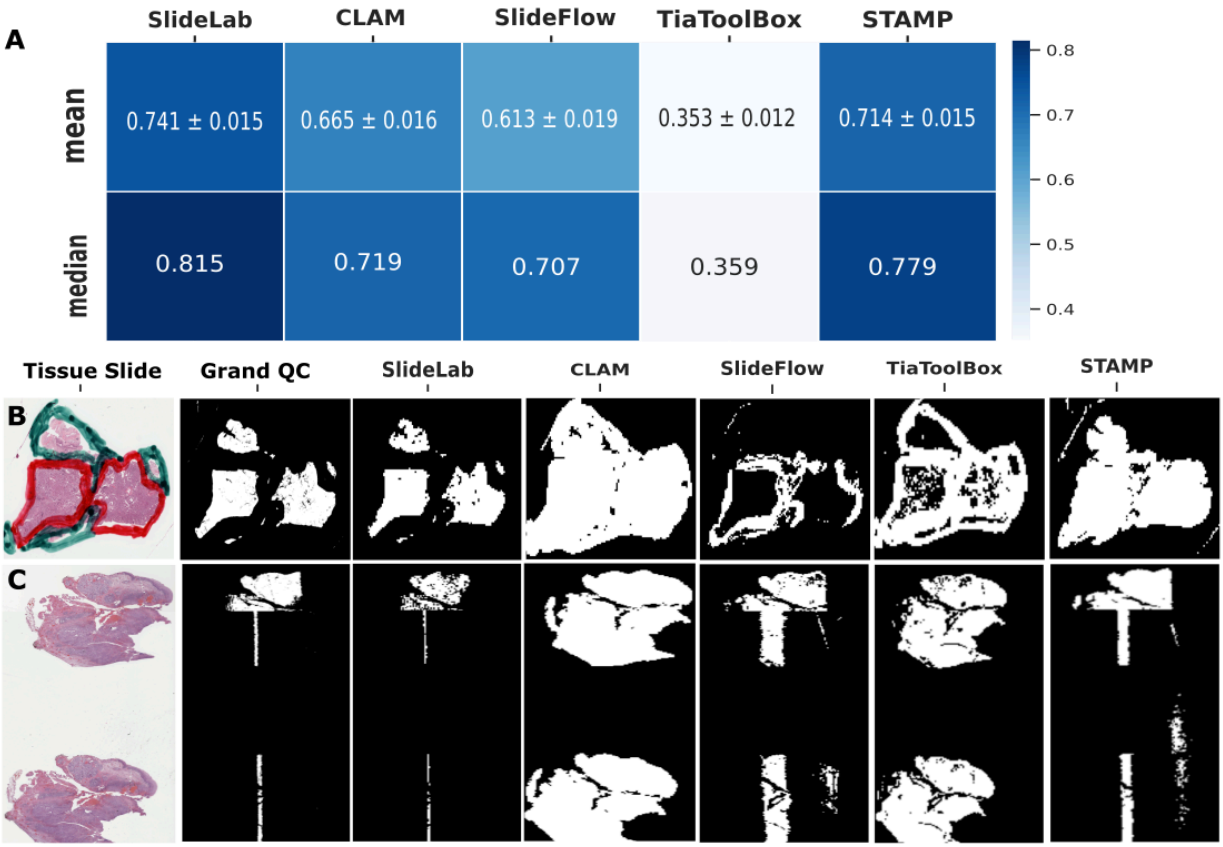


**Figure 1.10: Benchmarking SlideLab's Tissue Segmentation Performance.**

*This figure presents a quantitative and qualitative comparison of SlideLab's tissue segmentation against several other frameworks, including CLAM [15], SlideFlow [51] , TiaToolBox [53] , and STAMP [35]. The performance is measured against a ground truth provided by GrandQC [26]. The top panel (A) shows a heatmap of the mean and median Dice scores for each framework, where a score of 1 indicates a perfect match. The bottom panels, (B) and (C), provide a visual, qualitative comparison by displaying the original whole slide image and the corresponding binary segmentation masks produced by each framework. (B) Shows sample TCGA-86-8056-01Z-00-DX1, a slide with multiple pen marks which span both background and tissue. (C) Displays sample TCGA-IQ-7630-01Z-00-DX1, a slide that is severely out of focus.*

Qualitative examples, as exemplified in Figure 1.10B and 1.10C, support SlideLab's ability to exclude artifacts effectively. Its reconstructed slide masks closely mirror the GrandQC [26] reference. In contrast, tools like CLAM [15], SlideFlow [51] , and TiaToolBox [53] (Figure

1.10B) showed a tendency to oversegment regions or missegment artifact regions as the only "true tissue" regions.

While Dice scores provide a strong measure of overall, pixel-level accuracy, they can be misleading as they do not capture the quality of the final dataset from an instance level perspective. A high score can mask the inclusion of individual tiles that contain some portion of artifacts, which should be excluded to ensure sample quality. Therefore, to assess the purity of the final tile set, a more granular, tile-level analysis was performed.

### *1.4.2.2 Tile-Level Quality Assessment*

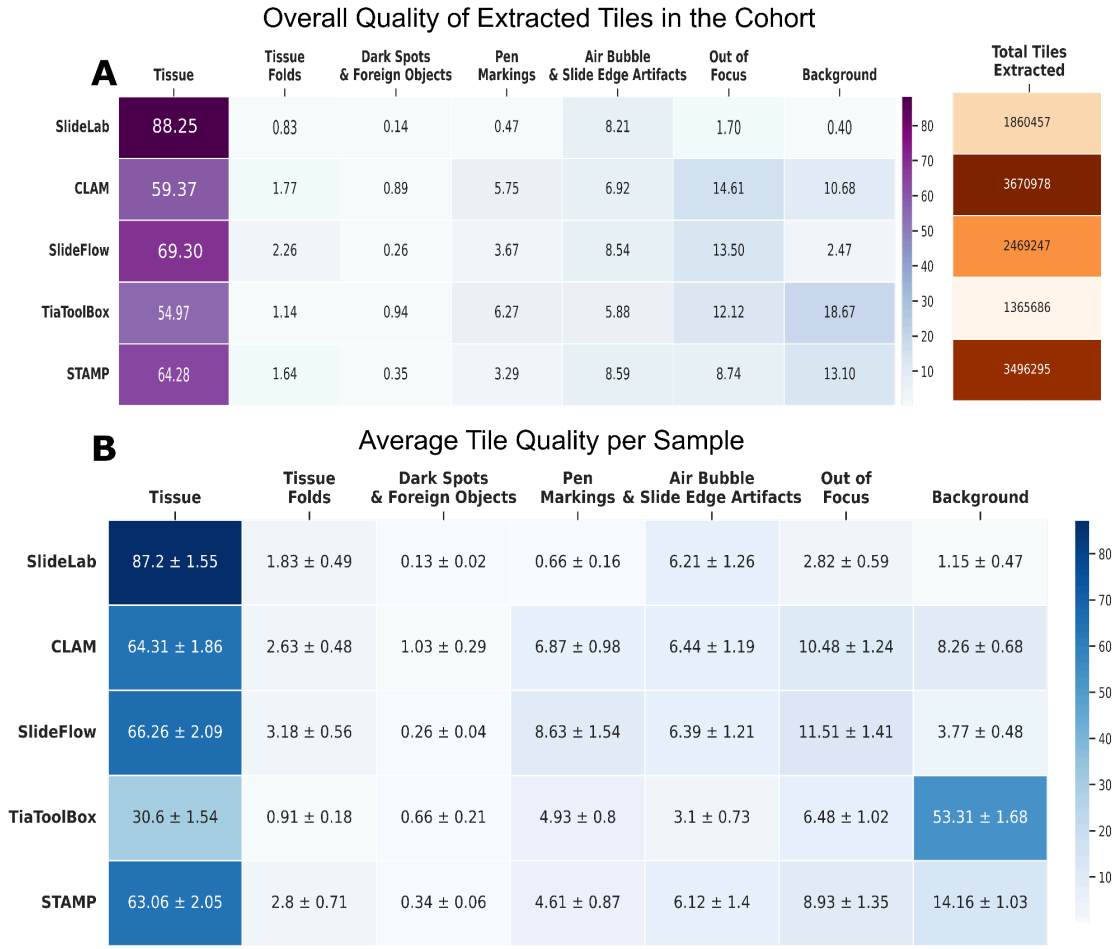

**Figure 1.11: Benchmarking SlideLab's Instance Quality Performance.**

*This figure presents a classification of the final tile cohorts for each benchmarked tool. Each tile was classified into one of GrandQC's [26] seven categories. (A) Represents the overall classification distribution throughout all extracted tiles in the cohort and displays the total amount of tiles obtained per tool. (B) Represents the average tile classification distribution (%) at the sample level.*

Across the entire cohort, SlideLab produced the highest proportion of true tissue tiles at 88.25% (Figure 1.11A). This was a significant improvement over all other tools, including CLAM [15] (59.37%), SlideFlow [51] (69.30%), STAMP [35] (64.28%), and TiaToolBox [53] (54.97%).

Furthermore, SlideLab was highly effective at excluding tiles with artifacts. It yielded the lowest percentage of out-of-focus tiles (1.70%), background tiles (0.4%), pen marks (0.47%), foreign objects (0.14%), and tissue folds (0.83%). The one artifact category SlideLab performed worse in was in detecting air bubble and slide edge artifacts, which is the one kind of artifact SlideLab doesn't have an algorithm to detect. Nonetheless, this was a huge improvement over other tools. For example, out-of-focus tiles constituted a large portion of the output from CLAM [15] (14.61%) and SlideFlow [51] (13.50%). Other frameworks like STAMP [35] and TiaToolBox [53] , had the greatest percentages of background artifacts—13.10% and 18.67%, respectively.

This trend of high-quality output from SlideLab was consistent at the individual sample level (Figure 1.11B). On average 87.2 ± 1.55% of the tiles in a given sample processed by SlideLab were classified as tissue. Most of the frameworks also had a persistent performance at the sample-level, with the exception of CLAM [15] (which improved) and TiaToolBox [53] (which worsened).

In summary, the combination of the highest Dice score for overall segmentation and the highest proportion of true tissue tiles—coupled with the lowest levels of artifact contamination—demonstrates that SlideLab had the best overall quality control performance, consistently producing higher quality datasets than other benchmarked tools.

## 1.4.3 Model Performance

To isolate the impact of preprocessing on downstream classification, an abMIL model was trained and evaluated for each tool's output. The primary measure of interest was the influence of preprocessing on the model's ability to generalize, which was assessed by its performance on the independent TCGA-COAD test set. All pairwise statistical comparisons were performed using the Wilcoxon signed-rank test, with an $\alpha$ of 0.05. Given the significant class imbalance in this task, metrics such as AUC, F1-score, Precision, and Recall provide a more insightful assessment of model performance than Accuracy; thus, they were prioritized in the subsequent analysis. Due to issues with processing larger cohorts, as mentioned in section 1.4.1, pathML [52] and TiaToolBox [53] were excluded from this evaluation.

### *1.4.3.1 Performance on the FFPE samples*

On the FFPE test set, models trained on data from both SlideLab configurations demonstrated an average superior performance. The model using SlideLab's full pipeline (quality control and stain normalization) achieved the highest mean F1-score (0.497 ± 0.136) and AUC (0.761 ± 0.108) (Figure 1.12.B). This performance was followed by SlideLab's quality control only configuration with a mean F1-score (0.482 ± 0.054) and AUC (0.733 ± 0.023). In comparison, the next best performing tools, SlideFlow [51] (AUC: 0.721 ± 0.02; F1: 0.386 ±

0.058) and STAMP [35] (AUC: 0.714 ± 0.012; F1: 0.327 ± 0.078) had a lower performance. For instance, when comparing their performance in these metrics to the baseline SlideLab quality control set up, the differences in the F1 value distributions were statistically significant for both SlideFlow [51] ($p = 0.002$) and STAMP [35] ($p = 0.0039$). For the AUC, this baseline displayed a significant difference in STAMP [35] ($p = 0.0469$) but not in SlideFlow [51] ($p = 0.0840$). CLAM [15] was the lowest performing tool in this task (AUC: 0.644 ± 0.029; F1: 0.294 ± 0.082), with all of its metrics showing a statistically significant difference from the best performing tools.

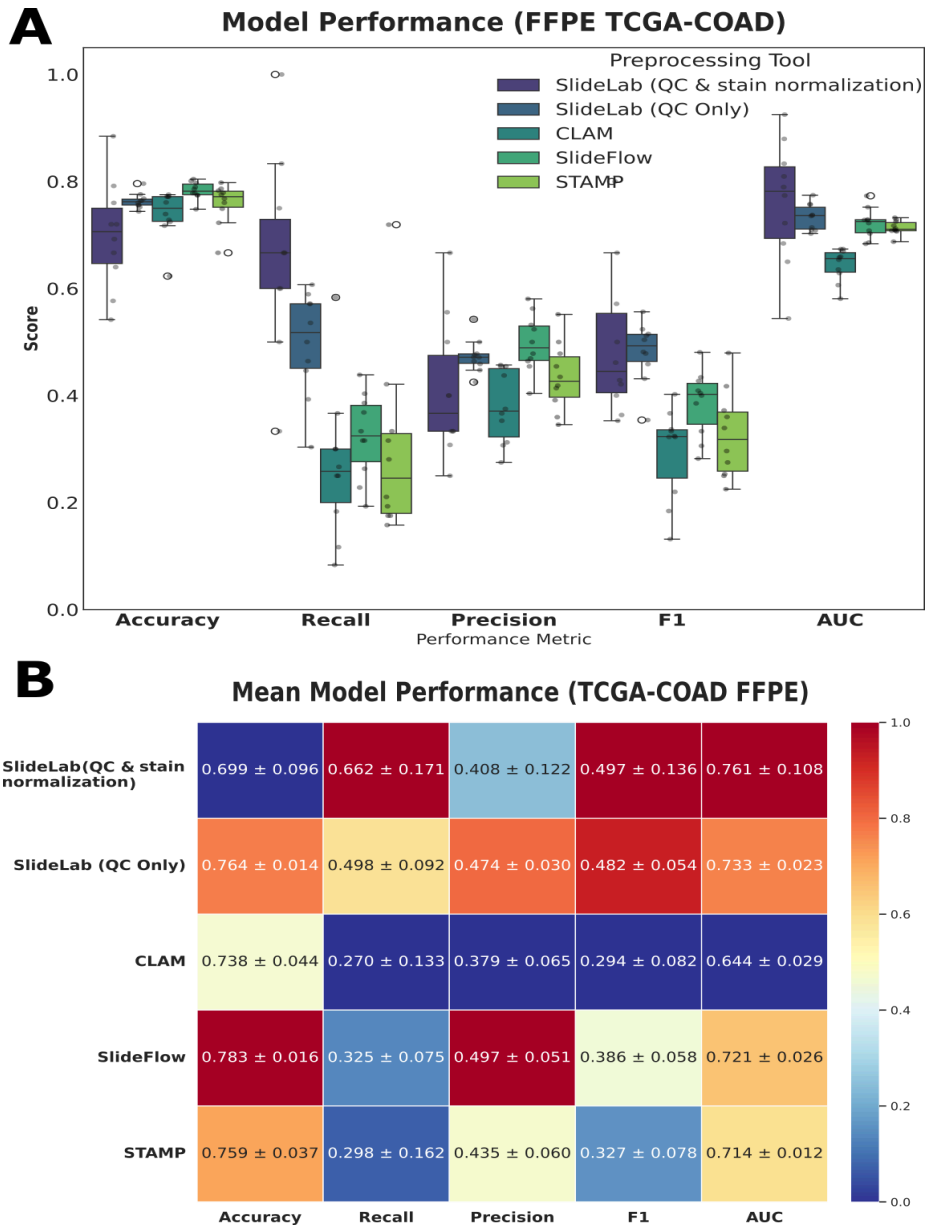


**Figure 1.12: Model performance in the TCGA-COAD FFPE test set.**

*(A) Boxplot of model performance across the 10-fold crossvalidation per preprocessing tool. (B) Mean model performance with standard deviation per model performance metric. The colors are normalized per column.*

When comparing it to the full SlideLab pipeline, this difference considerably widened for the F1 scores, with p-values of 0.0195 and 0.0273 for STAMP [35] and SlideFlow [51] , respectively. There was also a significant difference in the recall, showing a better ability to

identify true positive cases, reflected in a significantly higher mean Recall than CLAM [15] ($p=0.006$), SlideFlow [51] ($p=0.002$), and STAMP [35] ($p=0.002$) based models. Consequently, this led to a slight decrease in precision in comparison to other tools as the full SlideLab based model was more likely to classify instances as positive (MSI-H) than negative (MSS) (Figure 1.12B). Additionally, it is important to note, that in this instance the full SlideLab pipeline showed a greater standard deviation across folds throughout all metrics than the other pipelines and configurations, meaning that it was more unstable in its predictions than its quality control only counterpart (Figure 1.12A).

#### *1.4.3.2 Performance on the FF samples*

In this task, SlideLab's full pipeline was unequivocally the top performer achieving the highest mean F1-score ($0.460 \pm 0.030$) and AUC ($0.710 \pm 0.008$) paired with exceptionally low variance, indicating a stable and reliable model. This performance was statistically superior to CLAM [15], SlideFlow [51] , and STAMP [35] across key metrics, including Recall, F-1 score, and AUC (all $p<0.05$) (Figure 1.13B).

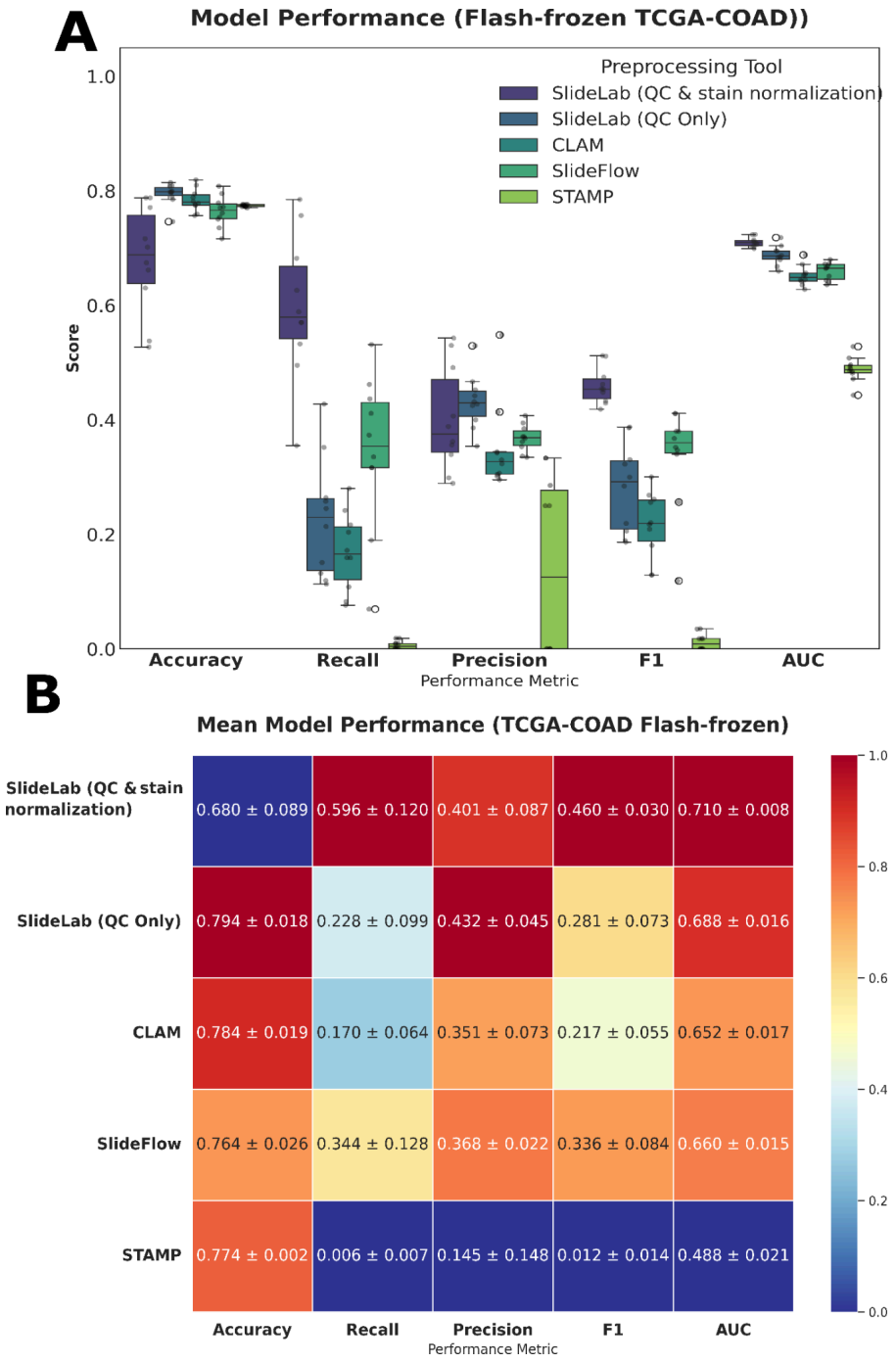


**Figure 1.13:** ***Model performance in the TCGA-COAD Flash-frozen test set.***

*(A) Boxplot of model performance across the 10-fold crossvalidation per preprocessing tool. (B) Mean model performance with standard deviation per model performance metric. The colors are normalized per column.*

Following the top-performing full SlideLab pipeline based model, the models trained on data from SlideLab configured for only quality control and SlideFlow [51] were competitively tied for second place. While the SlideFlow-based model [51] achieved a higher mean F1-score (0.336 ± 0.084 versus 0.281 ± 0.073) due to its higher mean recall, this difference was not significant ($p = 0.1602$) (Figure 1.13A, 1.13B) . The SlideLab (QC only) model was more precise and demonstrated a better underlying discriminatory ability with a higher mean AUC (0.688 ± 0.016 versus 0.660 ± 0.015; $p= 0.0059$).

The most critical finding was the failure of the STAMP-based [35] model. While its mean accuracy of 0.774 appears reasonable, this is merely an artifact of the class imbalance as it is nearly identical to the proportion of the majority class. Its mean F1-score of 0.012 and mean Recall of 0.006 demonstrate a near-complete failure to identify the minority class (MSI-H) when compared to SlideLab's full pipeline ($p = 0.002$ for both metrics). Much like in the FFPE task, the CLAM [15]-based model was one of the lowest performing models with an AUC of 0.652 ± 0.017 and F1 score of 0.217 ± 0.055 (Figure 1.13B).

Unlike in the FFPE task, the full SlideLab pipeline exhibited exceptionally low variance on the FF data, performing more consistently than its quality control-only counterpart. This strongly indicates that the instability observed on the FFPE set was not a framework-inherent issue but rather a dataset-specific effect. On the noisier FF data, stain normalization appears to act as a powerful standardizing force, reducing the visual heterogeneity from freezing artifacts and leading to a more robust and reliable model.

To summarize, the results demonstrate the clear superiority of the SlideLab framework for preparing WSI data for downstream analysis. In both its baseline quality control configuration and its full pipeline with stain normalization, SlideLab consistently produced tile

sets that led to more effective and generalizable models in CLAM [15], SlideFlow [51] , and STAMP [35]. The ablation study revealed that artifact removal does improve model performance, which is further enhanced by the utilization of stain normalization.

## 1.5 Discussion

Much like any data science task, the development of robust and generalizability in computational pathology is critically dependent on both the quality and standardization of the input data. However, the preprocessing of whole-slide images is often treated as a trivial step, with existing tools frequently representing a trade off between speed and functionality. This chapter introduced and benchmarked SlideLab, a comprehensive pipeline designed to bridge this gap by providing a solution that is computationally efficient, produces state-of-the-art data quality, and demonstrably improves downstream model performance in solid tumor tasks. The results from this three-pronged benchmarking approach provides several key insights into the current landscape of WSI preprocessing, with wider implications for model improvement independently of model architecture.

First, our findings highlight a significant disparity in the computational efficiency of available tools, especially given their reduced functionality. Many frameworks, such as CLAM [15], SlideFlow [51], and PathML [52], are much slower than SlideLab's simple tiling baseline. Furthermore, while tools like STAMP [35] offer exceptional speed and memory utilization, the techniques used to achieve these speeds are insufficient for producing analysis-ready data. This is seen in its model performance results. In contrast, SlideLab balances speed with quality. While slower than STAMP [35], it outcompetes all other benchmarked tools in CPU and user/wall time. Even with the additional steps of its full pipeline, it remains much more efficient than other frameworks, making large-scale, high-quality preprocessing computationally feasible for a wider range of researchers.

Second, the quality control benchmarks revealed that pipelines without robust artifact detection produce datasets that are heavily contaminated with information that is not

biologically-relevant. The effects of these were clearly seen specially in the flash-frozen MSS vs MSI-H prediction task, where models based on data preprocessed by CLAM [15] and STAMP [35] suffered greatly in performance; STAMP's output for example, was essentially useless for model training, yielding an F1-score of near zero. On the contrary, SlideLab was successfully able to outperform other tools in both the higher quality FFPE and lower quality Flash-frozen tasks with its simple quality control baseline and its more complex stain normalization and quality control configuration. This starkly illustrates that the quality of the tile set is far more important than the quantity, and that simplistic tiling and basic background filtering is not a viable strategy for building reliable models, especially on real-world, artifact-prone data.

Third, the ablation study provided by the two Slidelab configurations allowed us to quantify the impact of quality control and normalization on model performance. The quality control only based model demonstrated that artifact removal can significantly boost model generalizability. This with addition of stain normalization boosted model sensitivity (Recall), model precision (Precision), and proper distinction between classes (AUC and F1) for both FFPE and Flash-frozen cohorts. This insinuates that the inclusion of these denoising and data standardization steps can dramatically affect the power of the data.

However, this study was not without its limitations. For example, the quality control evaluation heavily relied on GrandQC [26] , a deep learning model, as the ground truth, which is itself subject to some degree of error. Future work could validate these findings against manually annotated masks. In addition to this, a main limitation of this study was actually limitations of some existing preprocessing frameworks used, with PathML [52] and TiaToolBox [53] having to be excluded from parts of this study due to the lack of capacity to preprocess larger datasets.

Future work on this subject should compare SlideLab's performance to a greater variety of frameworks with considerable computational efficiency.

Furthermore, this study focused on a single, albeit challenging, classification task. Future research should explore the impact of these preprocessing strategies on other downstream tasks, not limited to other biomarker predictions, survival analysis, and tumor grading.

Additionally, to make comparisons in computational efficiency fair, SlideLab's GPU capabilities were not compared to other pipelines. In a future study, it would be beneficial to see how this affects SlideLab's speed.

Finally, SlideLab could be improved by incorporating algorithms to detect the one class of artifact it currently misses: air bubbles and slide edges.

## 1.6 Conclusion

The preprocessing of WSI is a foundational step in the computational pathology workflow, yet it is often overlooked in favor of complex model architectures. Existing tools frequently force researchers to create their own pipelines due to their inefficient, inflexible nature.

This thesis chapter introduced SlideLab, a comprehensive, open-source pipeline that successfully increases flexibility. Through a rigorous benchmarking study, we have shown that SlideLab successfully integrates improved computational efficiency with high-quality results, with a purer tile set than any other benchmarked tool.

Most importantly, this thesis chapter shows that this superior data quality and standardization directly translates into increased performance in downstream model performance tasks. By effectively removing artifacts and standardizing stain profiles, SlideLab enables the training of more accurate, stable, and generalizable models.

## 1.7 Supplemental Material

### 1.7.1 Data and Code Availability

The code repository for the SlideLab project can be found at https://github.com/lolmomarchal/SlideLab . The WSI from the TCGA cohort are available from https://portal.gdc.cancer.gov/. The quality control masks generated by GrandQC [26] for TCGA are deposited at Zenodo under the following accession code (https://zenodo.org/records/14041578). MSI status for the TCGA-COAD cohort is available at https://github.com/KatherLab/cancer-metadata/blob/main/tcga/liu.xlsx. The Mutographs dataset is currently not publicly available, however weights for the trained models can be found in: https://drive.google.com/drive/folders/1h9BpjzI-Vebu8hJp-q0ztvQ2sCiwvcvp?usp=sharing.

# Chapter 2 Interpretability: Polarized Attentional Confidence

## 2.1 Introduction

The rapid development of artificial intelligence has directly impacted the growth of computational pathology as a transformative discipline in modern medicine and biomedical research.. Yet, despite its potential, the successful integration and overall utility of AI are contingent upon its transparency and interpretability.

In clinical practice, one of the main limitations regarding the implementation of AI-driven workflows is establishing trust and accountability in AI-driven decision-making. Given that most models provide limited insights into why they make decisions (i.e., they are "black boxes"), medical professionals like pathologists are hesitant to introduce them into their practices. This hesitation stems from the potential consequences such as patient harm and practitioner liability [20]. Similarly, in research, the limited interpretability constrains the ability to derive new biological insights into tasks like biomarker discovery, consequently leading to an underutilization of AI's potential in studying complex biological relationships and features [17][21].

Furthermore, model interpretability is a highly important feature from the regulatory perspective. Guidelines such as the "Good Machine Learning Practice for Medical Device Development: Guiding Principles" from the Food and Drug Administration (FDA) in the USA and the Medicines and Healthcare products Regulatory Agency (MHRA) in the UK have provided an initial generalized framework to build upon [65]. One of the main guidelines in this

document is the “emphasizing human-AI teamwork”, which refers to model interpretability for physician use and validation. While, currently, there are no specific regulatory pathways for AI in healthcare, most recent FDA approvals have relied on more generalized routes like the *510(k)* and *de novo* routes [66][67].

The *de novo* route in particular requires extensive and robust evidence of safety and effectiveness, which includes metrics like clinical validation, performance metrics, and explainability of the model to understand potential patient risk and product biases. Thus, there is a clear need for interpretability of models [68].

In addition to this, there is an ethical component to the need for interpretability of models. Much like any kind of AI task, the diversity of the training set directly impacts the biases a computational pathology model may learn. A recent study published in 2024 found that state-of-the-art computational pathology models tend to have a demographic bias, with larger disparities in misclassification occurring for black, hispanic, and indigenous patients due to underrepresentation in clinical datasets [69]. This is incongruent with current medical needs. For example, black people in the USA have the highest death rate for cancer overall with a greater chance of being diagnosed with breast, lung, and colorectal cancers at a later stage than other groups [70]. Given one of the main arguments towards computational pathology is early detection and medical barrier reduction through reducing the need for molecular testing, being able to uncover these types of biases are essential for AI model delivery [71].

### 2.1.1. The challenge of Interpretability in Computational Pathology

Model interpretability is the degree to which a human can understand the cause of a model's prediction. In computational pathology, this is distinctly difficult due to the reliance on complex deep learning algorithms. Many tasks in computational pathology are framed as Multi Instance Learning (MIL) problems, which are a form of weak supervision tasks. In this paradigm, a whole-slide image (WSI) is treated as a "bag" of smaller image patches or "instances," but only the entire bag has a label (e.g., "malignant" or "benign") [22][29][35]. The individual instances within the bag are not labeled, instead, it is the model's task to learn what specific instances in the bag are responsible for the label. This weak supervision creates a significant challenge, as the model's final prediction is based on a complex aggregation of information from many unlabeled instances.

A variety of methods have been developed to gain some understanding of the "black box" nature of deep learning models. These methods are typically categorized as post-hoc or intrinsic. Post-hoc methods, including Saliency Maps [72], SHAP [73], and LIME [74], are model-agnostic techniques applied after training. They generate explanations by examining how perturbations to a model's input features affect its output. In the context of computational pathology, a major limitation of these methods arises from model architecture. The SHAP and LIME methods are primarily designed for models with a fixed number of input features, which is not the case in a typical MIL framework where a "bag" contains a variable number of instances. This makes it difficult to directly apply their core permutation-based algorithms. Saliency maps, although more robust, have been consistently seen as "less trustworthy" in the literature for biomedical imaging as studies have shown that even with random model parameters, saliency

methods can still produce plausible-looking, but ultimately meaningless, heat maps or produced poorly-localized results [75][76]

Currently, the most prominent solution for providing interpretability in MIL models is the inclusion of an attention mechanism [77]. Attention-based MIL (abMIL) models assign an attention score to each instance in the bag , which represents the instance's contribution or weight to the final prediction (Equation 1.1) [22]. Previous studies [15][35] have used this to create a heatmap of important regions on individual WSI. However, even with this mechanism we are left with two distinct sources of information from the same model per tile: the prediction score (a measure of overall class likelihood) and the attention score (a measure of how much importance the model is giving an instance for the final decision).

$$a_k = \frac{\exp\left(w^T \tanh(V h_k^T)\right)}{\sum_{j=1}^{K} \exp\left(w^T \tanh(V h_j^T)\right)} \tag{1}$$

- $a_k$: Attention score for the $k$-th instance.
- $h_k$: Feature vector (embedding) for the $k$-th instance.
- $V, w$: Learnable weight matrices/vectors of the attention network.
- $K$: Total number of instances in the bag.
- $\tanh(\cdot)$: Hyperbolic tangent activation function (non-linearity).
- $\exp(\cdot)$: Exponential function.

**Equation 2.1. Attention scoring**

Unlike the prediction score, attention scores are non-directional measurements, as they are simply a weighing mechanism that highlights regions of importance. They do not, by themselves, reveal whether a high-attention region contributes positively or negatively to the final prediction. For instance, a high-attention score could be given to a region that provides

strong evidence for the target class (e.g., malignant cells) or to a region that provides strong evidence for the negative class (e.g., benign cells), and attention maps alone would not differentiate between the two. Similarly, looking at simple probability maps would only provide information on the model's overall confidence per instance, without revealing what features or regions of the cell the model paid attention to arrive at that score. This leaves a significant gap in our understanding of what the model is truly "seeing."

The literature attempted to address this with methods such as Prediction-Attention-Weighted (PAW) maps [78]. PAW maps represent a step forward by attempting to subdivide high-attention regions into positive and negative evidence. However, this method relies on filtering, setting an attention threshold and only considering the prediction scores of the most highly-attended tiles [78]. Through this, a large portion of the data, which may provide nuanced insights as to *why* they are considered important, is discarded. Thus, this approach fails to provide a single, comprehensive metric that quantifies both the degree of attention and the directionality for every instance.

### 2.1.4. Our solution: Polarized Attentional Certainty (PAC) Score

To overcome these limitations and provide a more robust and clinically relevant metric, we introduce the concept of Polarized Attentional Certainty (PAC). Unlike prior methods, PAC is an approach that combines the certainty of its prediction (via the log odds of the raw logits) and localized attention scores of the individual instances. By multiplying these two scores after normalizing them, PAC generates a single, comprehensive value that ranges from -1 to 1. This directly represents both the degree to which the model is weighing a specific instance and whether that instance provided strong negative or positive evidence for classification. From this

score, improved visualization and instance-level analysis can be done to understand and communicate model results.

The following sections will detail the implementation of PAC and showcase two separate application scenarios. The first case is using PAC to analyze the tumor microenvironment within extreme model decisions in a Microsatellite Unstable (MSI-H) versus Microsatellite Stable (MSS) task utilizing a multi-stage analysis  and nuclei morphometric analysis approaches. The second task is analyzing cell immunophenotypes of extreme cells in a B-cell Acute Lymphoblastic Leukemia (B-ALL) versus non-B-ALL task through direct quantification of cell marker intensities obtained through Flow Cytometry (FCM) .

## 2.2 The Tumor Microenvironment in Microsatellite Unstable Colorectal Tumors

### 2.2.1 Biological Context

Microsatellite Instable Hot (MSI-H) and Microsatellite Stable (MSS) are biomolecular markers present in a variety of cancers, including colorectal and endometrial tumors [79]. This distinction hinges on the functionality of the DNA mismatch repair (MMR) system [80][81][82]. An MSI-H status signifies a strongly deficient MMR system within the tumor cells, where key caretaker genes responsible for mismatch correction like MLH1, MSH2, and MSH6, are defective or suppressed [79]. In healthy cells with a functional MMR mechanism, errors that inevitably arise during DNA replication are immediately identified and rectified [79]. However, in MMR deficient cells, these mutations are not detected and become permanent. For example, some studies in E. Coli. suggest that MMR deficiency can increase the rate of mutation by 150-fold [83].

Consequently, in MSI-H tumors the loss of this critical repair mechanism triggers a cascade of genomic consequences. These tumors tend to have a high tumor mutational burden (TMB), with mutations often consolidated in microsatellites (short, repetitive DNA sequences) leading to frameshift mutations that result in the production of numerous truncated and non-functional proteins [84]. These tumor-specific proteins, also known as neoantigens, are recognized by the innate immune system as “foreign”, resulting in a powerful immune response and activation of the adaptive immune system [84][85]. When innate cells like antigen presenting cells encounter these neoantigens, they release signaling molecules called cytokines, which in turn summon adaptive immune cells like killer-T cells to the tumor location (Figure 2.1) [85][86].

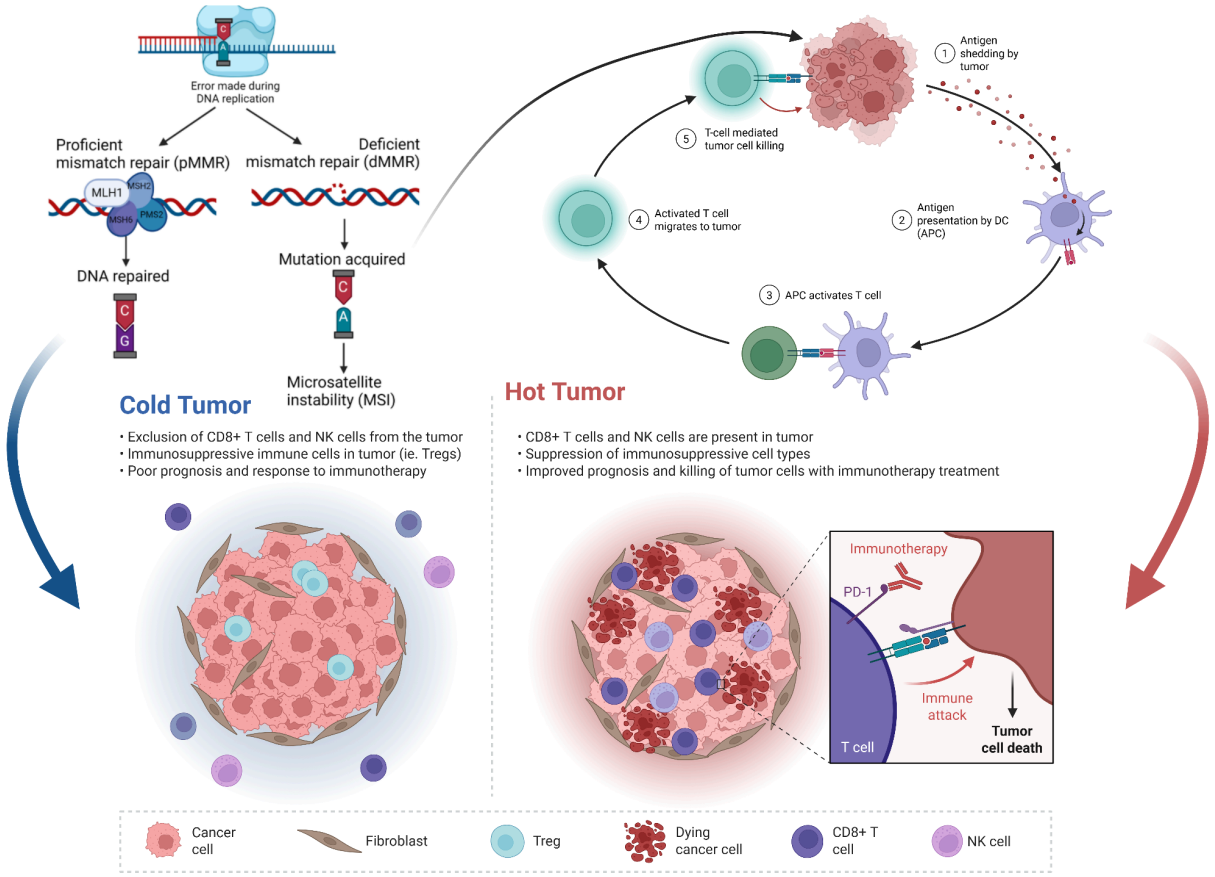


**Figure 2.1: The effect of mismatch repair deficiency on the tumor microenvironment.**

*Tumors exhibiting proficient mismatch repair mechanisms typically manifest as "cold tumors" characterized by a limited immune cell presence. Conversely, mismatch repair deficient tumors tend to release an increased quantity of neoantigens, consequently presenting as "hot tumors" with elevated immune infiltration.*

Because of this, MSI-H Tumor Microenvironments (TME) are highly immunogenic, with high infiltration of immune cells, including cytotoxic T lymphocytes and other effector cells [86][87]. The result is an inflamed or “immunologically hot” TME. In stark contrast to MSI-H tumors, MSS tumors do not suffer from MMR deficiency and thus do not produce such an excessive amount of neoantigens. Consequently, the immune activity in these tumors tends to be

low, leading to MSS tumors being characterized as "immunologically cold" or immune deserts [85][88][89].

The differing nature of these immune environments is also differentiable by their necrotic nature. MSI-H tumors, for example, exhibit significantly less "dirty necrosis" (a form of cell death marked by a heavy neutrophil infiltrate), further distinguishing their T-cell-rich inflammation from that of MSS tumors [90].

### 2.2.2 Case Study Background

The classification of colorectal tumors as either MSI-H or MSS is a foundational task in computational pathology, representing one of the earliest successful applications of deep-learning to histology slides (Kather et. al., 2019). Unlike more recent tasks, there is a significant body of research on this topic, driven by its therapeutic implications: MSI-H status is a key biomarker that predicts a strong response to immune checkpoint inhibitor therapy due to its immunogenic nature[85][88][89].

Accordingly, this task has high clinical stakes given the profound impact on patient treatment. MSI-H cases only represent 15-20% of the total global cases of colorectal cancer, however their therapeutic interventions differ greatly from MSS cases [61]. Research has consistently shown that MSI-H tumors are resistant to standard 5-FU-based chemotherapy. In the adjuvant setting for Stage II-III cancer, studies show these patients gain no survival benefit from 5-FU, a treatment that is highly effective for MSS patients [91][92]. This chemoresistance is attributed to come from MMR deficiency, which prevents the drug from triggering cancer cell death [93]. Conversely, immune checkpoint inhibitor therapy is not effective in MSS.

Because of this, understanding why a model makes a specific prediction is crucial for building clinical trust and ensuring its decisions have a verifiable biological basis.

## 2.3 Immunophenotype of B-cell Acute Lymphoblastic Leukemia

### 2.3.1 Biological Context

B-cell Acute Lymphoblastic Leukemia (B-ALL) is an aggressive cancer of the blood and bone marrow characterized by the overproduction of immature lymphoblasts, a type of white blood cell [94]. The main distinction between these malignant cells and normal B-cell development hinges on their patterns of protein expression, known as immunophenotype, which is defined by the tightly regulated process of B-cell mutation (hematopoiesis) [94][95]. In the healthy bone marrow, progenitor lymphoblastic stem cells mature into functional B-lymphocytes by systematically gaining and losing specific cell surface antigens at defined stages [95]. For example, early precursors express the stem cell marker CD34 [96], which is lost as they mature, while the pan-B-cell markers like CD19 and CD22 [97] are expressed throughout most of the lineage (Table 2.1).

**Table 2.1: Description of the cellular markers used in the B-ALL case study and their usual relative expression values in B-ALL.**

| | FSC | SSC | CD66B | CD22 | CD19 | CD24 | CD10 | CD34 | CD38 | CD20 | CD45 |
|---|---|---|---|---|---|---|---|---|---|---|---|
| **What** | Size | Granularity | Granulocyte cell marker | Pan B-cell marker | Pan B-cell Marker | Immature B-cell marker | Acute Lymphoblastic leukemia marker | Stem cell marker | Immature B-cell marker | B-cell marker | Leukocyte Common Antigen |
| **Expression in B-ALL** | Low to moderate | Low to moderate | Low to moderate, Higher than in B-cells | Moderate to high | High | Higher in B-ALL blasts than in B-cells | Very High | Very High | Very high in comparison to B-cells | Low to moderate | Low |

Consequently, in B-ALL, this maturation process is genetically disrupted and arrested. Essentially, this soft-tissue tumor is composed of a clonal population of lymphoblast “frozen” at an early developmental stage [98]. This arrest results in what is known as Leukemia-Associated Immunophenotype (LAIP), an aberrant combination of markers that does not typically correspond to any normal stage of B-cell maturation [98][99].

These blasts typically co-express very early markers like the stem cell antigen CD34 and the leukemia marker CD10, alongside B-lineage markers like CD19 and immature B-cell markers like CD24 . This asynchronous expression is the biological hallmark of the disease (Table 2) [99].

### 2.3.2 Case Study Background

The automated classification of leukemias via multi-parameter flow cytometry (FCM) is a well-studied use case in hematopathology, offering a rapid diagnostic tool critical for acute leukemias, which are highly aggressive and progress fast [100][101]. Unlike many cancers, B-ALL and other acute leukemias have no staging system; clinical guidelines focus on three main categories: untreated, in remission, or recurrent [102]. Because the disease progresses so quickly, diagnosis speed  and subclassification is paramount to patient prognosis.

This case study was selected as a parallel to computational pathology due to their similarities as near single-cell data. At its core FCM provides high-dimensional feature vectors for individual cells, which is conceptually analogous to the patch-based analysis of whole-slide images. This similarity in data structure has led to methodological crossovers. For example, recent publications in automated gating for FCM have begun to use abMILs due to the advantages of using a weakly-supervised approach for tasks such as AML classification [103][104]. Furthermore, this case was chosen because its features are simpler to analyze directly, and thus show the generalizability of PAC for tasks outside of computational pathology.

## 2.4 Polarized Attentional Certainty (PAC)

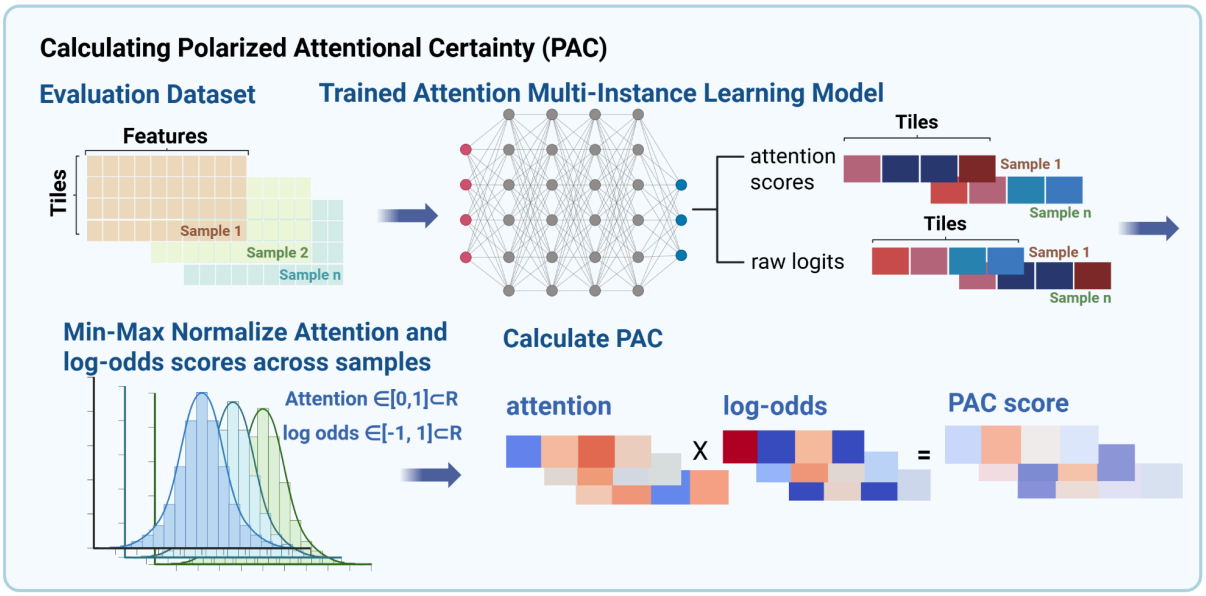


**Figure 2.2: Calculating Polarized Attentional Certainty (PAC)**

*PAC is calculated by obtaining the attention scores and the log-odds of the logits for each instance in some sort of evaluation set. These are obtained by putting instance features though a trained model that has an attention mechanism. The combination of the log-odds and attention results in the PAC score.*

The following subsections introduce the methodology for Polarized Attentional Certainty (PAC). The PAC score for a given instance is a composite of two distinct components: an attention score ($A_i$), which quantifies the weight given to it by the model for model decision, and a log-odds score ($C_i$), a measurement of the model's predictive certainty (Figure 2.2).

### 2.4.1 Calculating PAC

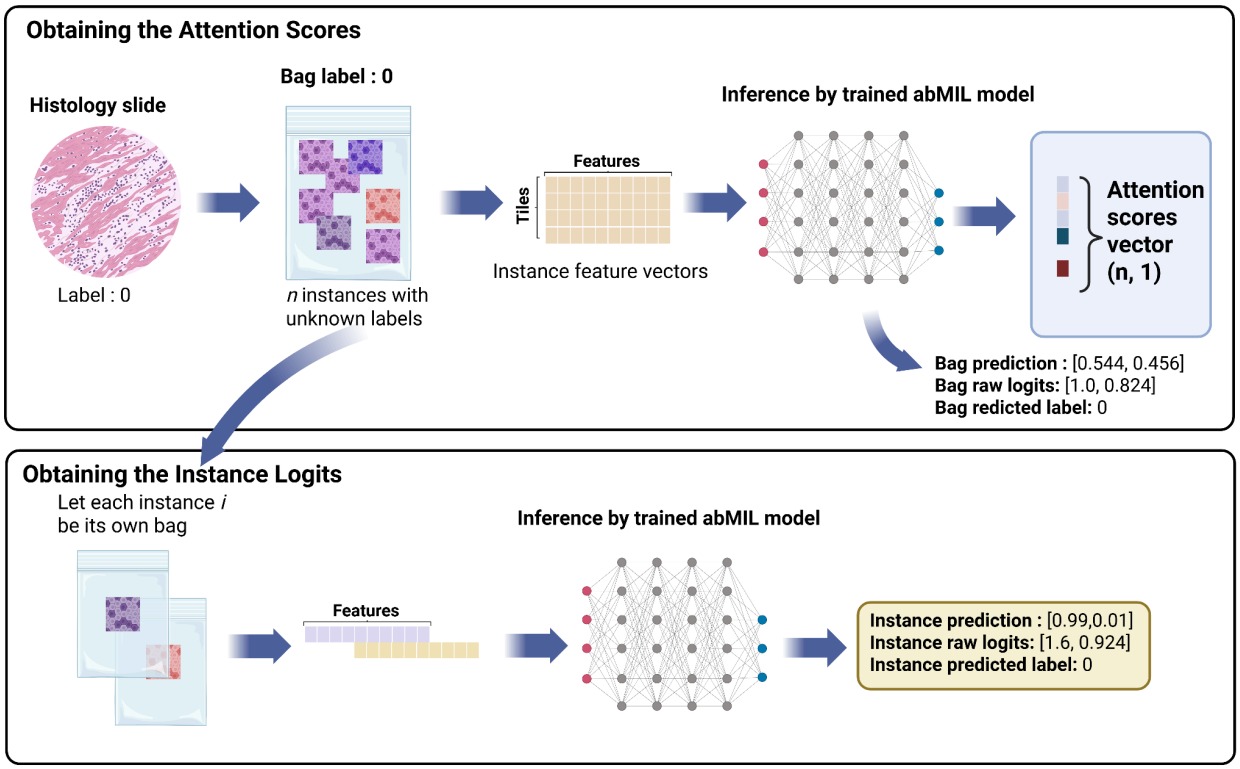


**Figure 2.3: Obtaining instance Attention and Raw logits.**

*This figure describes how to get the instance information needed to calculate their corresponding PAC score. The top row shows how the attention scores are obtained and the bottom represents how the instance-level logits are formulated.*

#### 2.4.1.1 Attention score

The attention score (*Equation 2.1*) for an instance *i* is derived directly from the model's attention mechanism after processing a sample (a full bag of instances with some bag size *K*). Essentially, it represents the learned weight or contribution of an instance *i* based on its feature vectors to the aggregate bag-level prediction, which is normalized across the bag (Figure 2.3).

#### *2.4.1.2 Log Odds of the Logits*

PAC utilizes the log-odds formulation (*Equation 2.2*) of the instance belonging to the positive class. This provides a measure of how much evidence there is that an instance belongs to the positive class.

In this structure, each instance $x_i$ is treated as a singleton bag and passed through the trained network to obtain the raw, pre-activation outputs (logits) for the negative class ($L_{i,0}$) and the positive class ($L_{i,1}$) (Figure 2.3). The log-odds score $C_i$, is then defined as the difference between these logits.

$$C_i = L_{i,1} - L_{i,0} \quad (2)$$

**Equation 2.2: Log odds score**

This log-odds formulation was deliberately chosen over post-softmax probabilities to avoid the potential saturation effect caused by its non-linear compression. Unlike probabilities, which can compress large differences in evidence into minuscule numerical changes as they approach 0 or 1, the unbounded nature of logits preserves this information. This allows the log-odds to directly represent the magnitude of the model's certainty [105][106].

For example, consider two instances that produce logits of [0, 10] and [0, 5], resulting in log-odds of 10 and 5, respectively. The evidence for the first instance is twice as strong. However, applying the softmax function (Equation 2.3) transforms these values into probabilities of approximately 0.99995 and 0.993. The large difference in evidence is truncated due to the non-linear compression. Essentially, using the log-odds preserves this magnitude of model certainty in neural networks [106].

$$P_{i,j} = \frac{e^{L_{i,j}}}{\sum_{k=1}^{K} e^{L_{i,k}}} \quad (3)$$

- $P_{i,j}$: The calculated probability of instance $i$ belonging to class $j$.
- $L_{i,j}$: The raw logit score for instance $i$ for class $j$.
- $K$: The total number of classes in the classification task.

**Equation 2.3. Softmax activation**

#### 2.4.1.2 Calculation of Polarized Attentional Certainty (PAC)

To standardize the scales of the log-odds and attention scores, we apply min-max normalization across all instances in the evaluation dataset. The attention scores ($A_i$) are normalized to a range of [0,1] (*Equation 2.4*), while the log-odds scores ($C_i$) are normalized to range of [-1,1] to preserve their directional interpretation (*Equation 2.5*).

$$A'_i = \frac{A_i - \min(A)}{\max(A) - \min(A)} \quad (4)$$

**Equation 2.4. Min-max normalized attention**

$$C'_i = 2 \cdot \frac{C_i - \min(C)}{\max(C) - \min(C)} - 1 \quad (5)$$

**Equation 2.5. Min-max normalized log-odds**

The final PAC score is then the product of the normalized attention ($A_i'$) and the normalized log-odds ($C_i'$) (*Equation 2.6*).

$$\boxed{\mathrm{PAC}_i = A_i' \times C_i'} \qquad (6)$$

**Equation 2.6. PAC score**

## 2.5 Applications Methodology: PAC-driven Instance Analysis

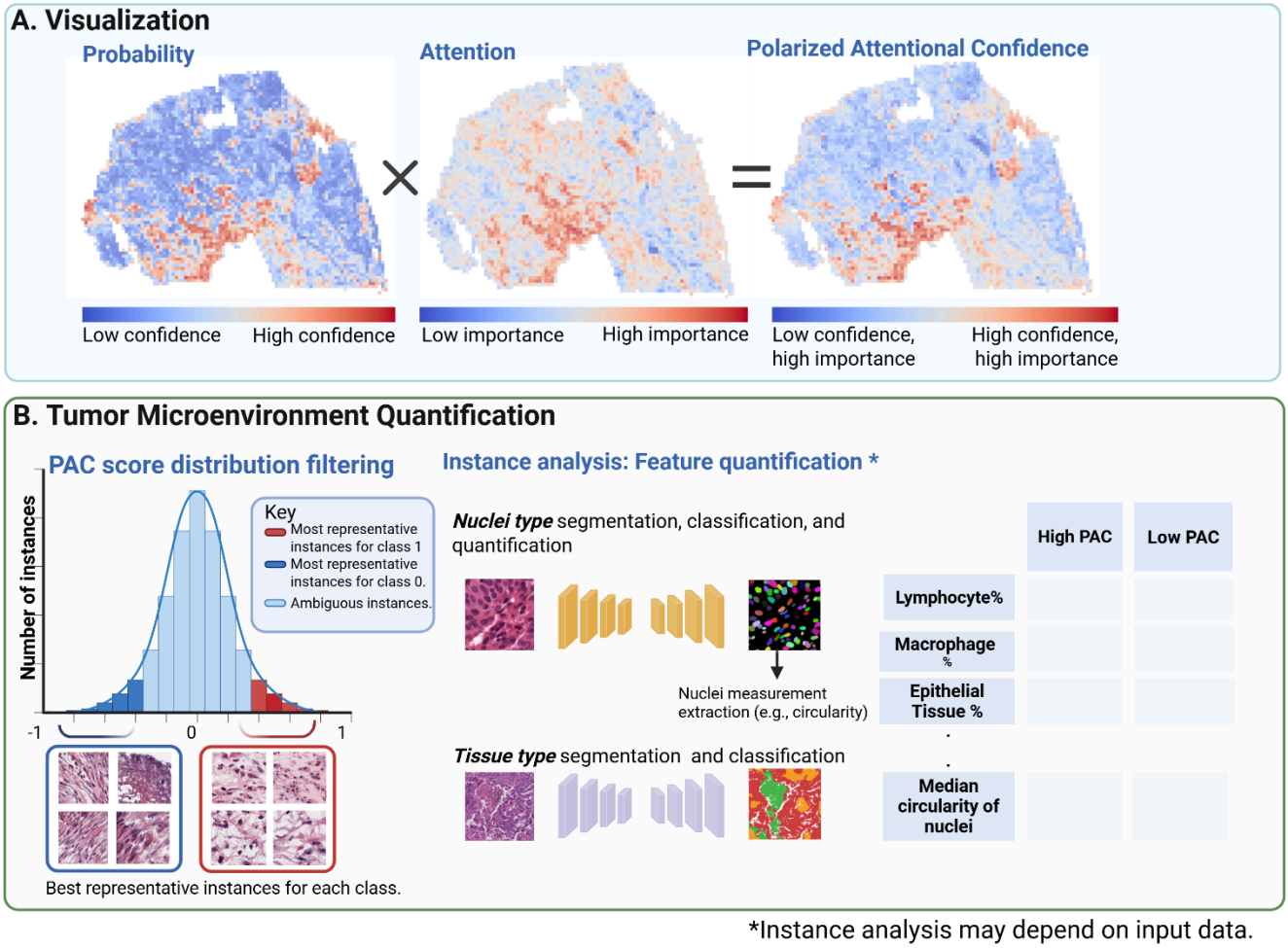


**Figure 2.4. Applications of PAC for Visualization and Model Decision Interpretation.**

*This figure exemplifies two broad categories that PAC can be applied towards. (A) Visualization heatmaps of instances by showing areas that the model is using to define class phenotypes. (B) Using PAC's distribution to analyze "extreme" instances that are representative of each class in the task. Once each extreme cohort has been selected, downstream analysis can be done on those instances to their metrics diverge and understand the weakly-supervised model's definition of the two classes. This analysis step is flexible depending on the input data.*

This section will detail the practical application of PAC as a tool for model interpretation and visualization. The general strategy used was utilizing the PAC score distribution to identify and isolate the most influential instances within the model's decision-making process. This was

done to essentially analyze what were the characteristics for the most representative instances of one class versus the other (Figure 2.4B). This framework's functionality is demonstrated across two distinct biomedical classification tasks.

### 2.5.1 Instance Cohort Selection

For each application, an abMIL model was first trained and evaluated in a cross validation scheme. The fold with the highest performance on an independent testing set, as measured by the AUC, was selected for the interpretability analysis. Following model selection, PAC scores were calculated for all instances in the evaluation dataset.

To identify the most representative instances for each class, we established two extreme cohorts based on percentiles. For example, the high-PAC cohort consisted of instances in the 99th percentile of PAC scores, representing the most representative instances for what the model interpreted as the positive class. Meanwhile the low-PAC cohort consisted of instances in the 1st percentile, representing the model's strongest interpretation of the negative class. These two cohorts formed the basis for all subsequent downstream analyses.

### 2.5.2 Case Study 1: The Tumor Microenvironment in Microsatellite Unstable Colorectal Tumors

#### 2.5.2.1 Model and Dataset information

This case study was done using the best performing model from the Flash-frozen (FF) task detailed in Chapter 1. This model was trained to classify Microsatellite Instability High (MSI-H) versus Microsatellite Stable (MSS) status in colorectal cancer from histology images.

The model that achieved the highest AUC on the TCGA-COAD FF task (0.82) was selected for this analysis.

#### 2.5.2.2 Analysis Information

##### 2.5.2.2. Foundational model analysis

To characterize the biological patterns within the extreme instance cohorts, a multi-stage analysis was performed using pre-trained foundational models from the TiaToolBox [53] framework. This approach allowed for a detailed compositional analysis of the tumor microenvironment.

An U-Net model [35][107] was first applied for tissue segmentation to assess the broader tissue context, categorizing regions under the following labels: Tumor, Stroma, Inflammatory, or Necrosis. For insights into cell-level populations, two variants of the HoverNet model [14] were employed for nuclei segmentation and classification. The HoverNet-PanNuke [14][35][108] model was used to obtain general cell type classification (connective, dead, neoplastic, inflammatory, non-neoplastic epithelial), while the HoverNet-MoNuSAC [14][35][109] model was used to obtain specific immune cell classifications (epithelial, lymphocyte, macrophage, and neutrophil).

##### 2.5.2.2.2 Nuclei Morphometric Analysis

To complement the classification data, morphometric features were calculated from the nuclei segmentation masks generated by the HoVerNet-PanNuke model. Features describing nuclear shape, such as eccentricity, area, and radial symmetry, were extracted. The distributions of these measurements were compared by calculating the mean and median values for the 1st and 99th percentile PAC cohorts. This analysis aimed to identify quantifiable morphological features that differ significantly between the cohorts predicted to be MSS and MSI-H.

### 2.5.3 Case Study 2: Immunophenotype of B-cell Acute Lymphoblastic Leukemia

#### 2.5.3.1 Model and Dataset information

The B-ALL prediction dataset consisted of 18 patients, 79 with a low proportion of B-ALL cells and 39 with a diagnostically high proportion of B-ALL cells. The testing set consisted of 24 patients, 10 of which had B-ALL. The Measurable Residual Disease (MRD) levels were obtained through manual gating by pathologists at UCSD Health, where the samples were collected. A standard cut-off of 0.001% MRD was used to stratify the samples into positive and negative cases [110]. An abMIL model was trained using a 5-fold cross-validation scheme and the best-performing fold, which achieved a testing AUC of 1.0, was selected for this analysis (Figure 2.5).

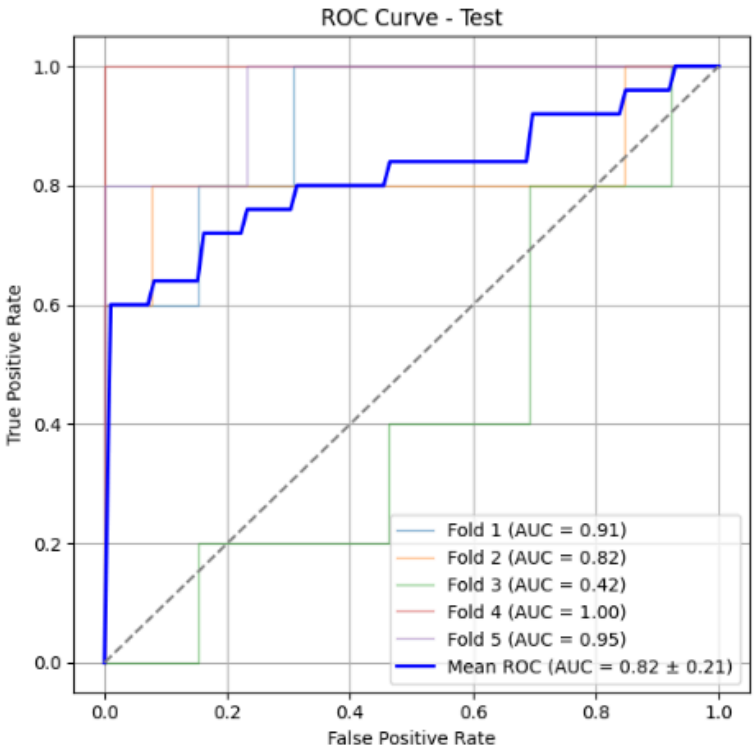


**Figure 2.5: Performance of the B-ALL classification model on the test set.**

#### 2.5.3.2 Analysis Information

Following the isolation of the extreme cell cohorts from the 1st and 99th PAC percentiles, a quantitative analysis was performed to determine the immunophenotype most associated with each class. This was achieved by calculating the median expression intensity for each cell marker in the tube for all cells within the testing cohort.

Given that the feature vectors for this case study were comparably lower in dimensionality than the MSI-H vs MSS task (11 features versus 2048) and had direct biological meaning, this application also included a comparative analysis to explicitly demonstrate the utility of PAC against other metrics. The 1st and 99th percentiles of cells were identified for both attention scores and probability scores alone. A comparison of these values against PAC was done.

### 2.5.4 Statistical Analysis

All of the statistical analyses were performed using the Mann-Whitney U test to compare the distributions of morphometric features between the high-PAC and low-PAC cohorts. This non-parametric test was specifically chosen because the data could not be assumed to follow a normal distribution due to cell type heterogeneity (in either the patches in case study 1 or cell populations in the FCM tube). An $\alpha = 0.05$ was used.

## 2.6 Results

### 2.6.1 Case Study 1: Tumor Microenvironment Characterization in MSI-H

#### 2.6.1.1 Interpreting PAC Maps in MSS and MSI-H Cases

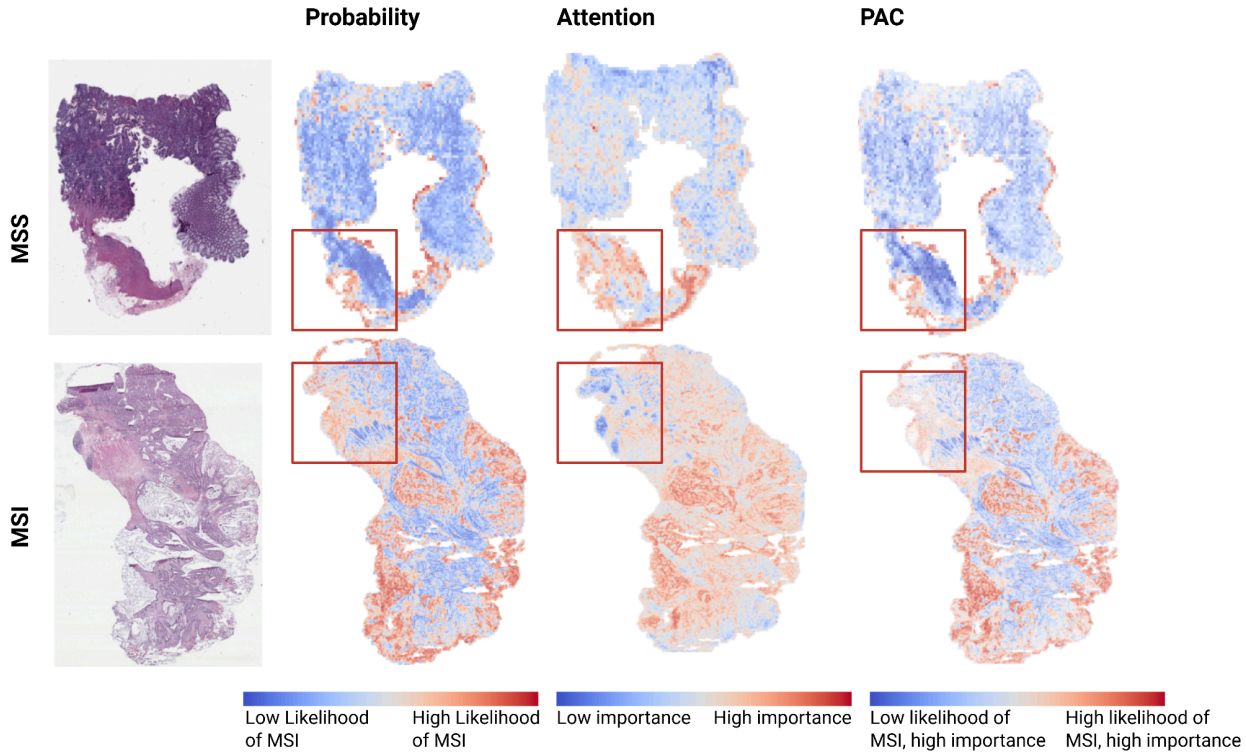


**Figure 2.6: Comparison of probability, attention, and PAC visualizations for representative regions of interest.**

*The top row showcases predictions for an MSS sample and the bottom row for an MSI-H sample. The red squares highlight regions of significant divergence between the metrics.*

Figure 2.6 demonstrates that probability and attention are often discordant in both negative (MSS) and positive (MSI-H) instances. In the top row (MSS case), the red-boxed region exhibits a very low probability score (indicative of an MSS prediction) but is assigned high attention. This suggests that, although the model predicts this region as MSS, it considers it highly informative for the classification. The PAC visualization combines these two aspects, highlighting the region as having high importance while supporting an MSS classification. This

can be interpreted as the region contributing significantly to the model's understanding of the negative (MSS) class.

In contrast, the bottom row presents an MSI-H case. The highlighted region shows a high probability score, indicating a strong MSI-H prediction; however, several areas within this region received low attention. In the PAC map, this appears as a near-white region, indicating a high likelihood of MSI-H but low overall contribution to the model's decision. This suggests that, although the region is predictive of MSI-H, it was not among the most influential features driving the model's definition of the MSI-H phenotype.

#### 2.6.1.2 Tissue and Cellular Composition Analysis

Analysis of around 3,489 image instances from each cohort revealed distinct differences in tissue morphology and cellular composition between the 1st percentile (MSS) and 99th percentile (MSI-H) PAC-scoring groups. The MSI-H cohort was characterized by a disorganized tumor architecture with prominent inflammatory infiltrates and extensive necrosis (Figure 2.7, bottom row). In contrast the MSS cohort exhibited more structured tumor cell clusters with dense, fibrotic stroma (Figure 2.7, top row), indicating a far less immunogenic tumor microenvironment.

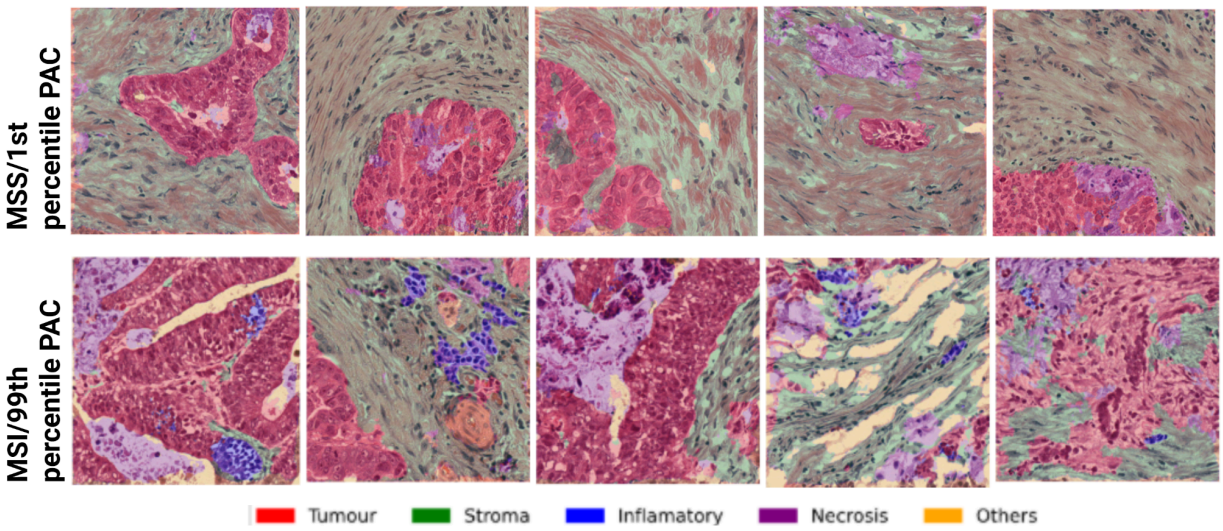


**Figure 2.7: Tissue segmentation results of a U-Net model overlaid on example tiles taken from the 1st and 99th percentiles of PAC scores.**

*The top row shows examples from the 1st percentile PAC tiles, representative of MSS. The bottom row shows examples from the 99th percentile of PAC instances, representative of MSI. The segmentation overlays highlight different tissue types: tumour (red), stroma (green), inflammatory cells (blue), necrosis (purple), and other tissue types (yellow).*

To quantify these morphological differences, we calculated the average area (in pixels²) of each tissue class within both cohorts (Table 2.2). This analysis confirmed that the MSI-H cohort (99th percentile PAC) contained, on average, significantly larger areas of inflammatory tissue (a 41.6x increase) and necrosis (a 7.7x increase) compared to the MSS cohort. Conversely, the MSS cohort (1st percentile PAC) was characterized by a predominance of stromal tissue, which aligns with its observed histological appearance (Table 2.2).

**Table 2.2: Mean tissue area (pixels²) per instance across PAC percentiles**

| AREA (pixel$^2$) | **Tumor** | **Stroma** | **Inflammatory** | **Necrosis** |
|---|---|---|---|---|
| **99th percentile PAC (MSI-H)** | 179437 | 133339 | 9868 | 178733 |
| **1st percentile PAC (MSS)** | 120568 | 670684 | 237 | 23273 |

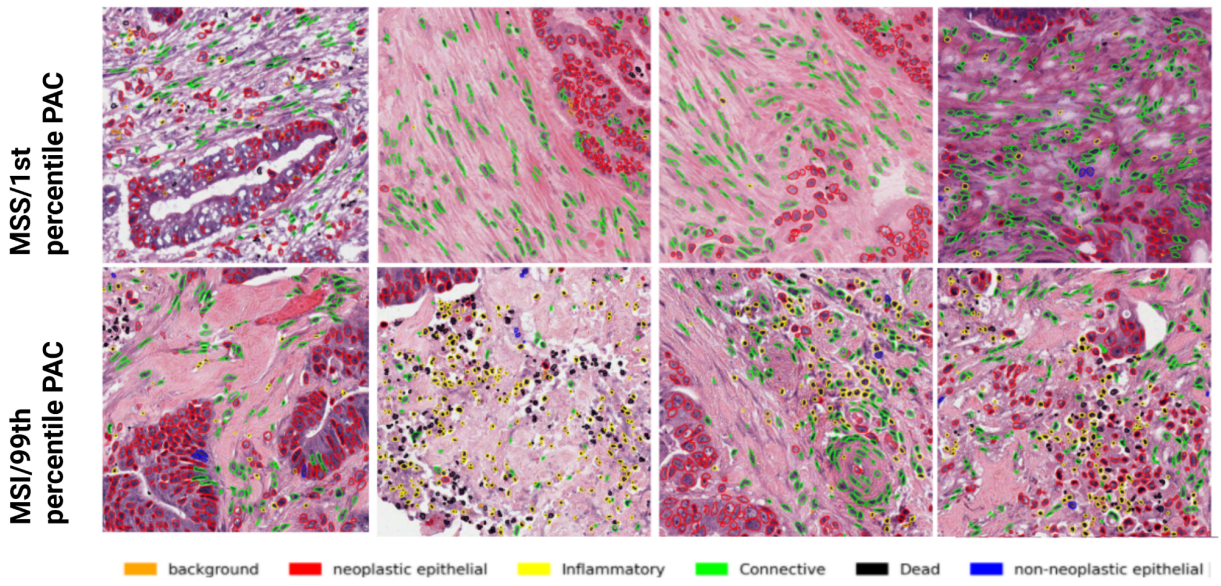


**Figure 2.8: Nuclei segmentation results from the pretrained HoverNet-PanNuke model overlaid on example tiles taken from the 1st and 99th percentiles of PAC scores.**

*The top row shows examples from the 1st percentile PAC tiles, representative of MSS. The bottom row shows examples from the 99th percentile of PAC instances, representative of MSI. The segmentation overlays highlight different tissue types: tumour/neoplastic epithelial (red), connective (green), inflammatory cells (yellow), necrosis (black), background (orange), and non-neoplastic epithelial (blue).*

To investigate the cellular composition in great detail, we performed single-cell analysis using the HoverNet model pre-trained on the PanNuke dataset to classify individual nuclei into biologically relevant types (Figure 2.8) [14][35][108]. The results revealed that the MSI-H cohort (99th percentile PAC) had a greater proportion of inflammatory nuclei (17.18% vs. 6.01%) and dead nuclei (11.61% vs. 3.12%; $p>0.0001$) compared to the MSS cohort. Conversely, the 1st percentile PAC cohort (MSS) was dominated by connective tissue-associated nuclei (48.99% vs. 27.18%; $p>0.0001$). These cellular findings directly reflect the tissue-level analysis, where a higher prevalence of connective nuclei corresponds to the extensive stroma in predicted MSS cases, while more inflammatory and dead nuclei align with the inflammation and necrosis seen in predicted MSI-H cases.

**Table 2.3: Nuclei classification by HoverNet model pretrained on PanNuke dataset**

| | Neoplastic | Inflammatory | Connective | Dead | Non-Neoplastic Epithelial |
|---|---|---|---|---|---|
| **99th percentile PAC (MSI)** | 41.04 | 17.18 | 27.18 | 11.61 | 1.12 |
| **1st percentile PAC (MSS)** | 39.60 | 6.01 | 48.99 | 3.12 | 0.66 |

In order to further characterize the inflammatory infiltrate, we used a HoverNet model trained on the MoNuSAC dataset [14][35][109] . This analysis revealed that most of the inflammatory infiltrate in the MSI-H representative set were composed of lymphocytes, which on average made up 41.66% of the detected immune cells, followed by smaller populations of macrophages (6.75%) and neutrophils (4.03%). In contrast, the MSS representative set was characterized by a significant proportion of epithelial-type cells (64.93%) and a correspondingly lower abundance of immune cells on average (Table 2.4).

**Table 2.4: Nuclei classification by HoverNet model pretrained on MoNuSAC dataset**

| | Epithelial | Lymphocyte | Macrophage | Neutrophil |
|---|---|---|---|---|
| **99% percentile PAC** | 37.946 | **41.66** | **6.75** | **4.03** |
| **1% percentile PAC** | 64.93 | 23.61 | 2.23 | 1.44 |

### 2.6.1.3 Nuclei Morphometric Analysis

The analysis of fundamental nuclear morphometric features revealed subtle but consistent differences between the extreme PAC cohorts, painting a distinct portrait of the typical nuclei found in representative MSI-H model predictions (Figure 2.9).

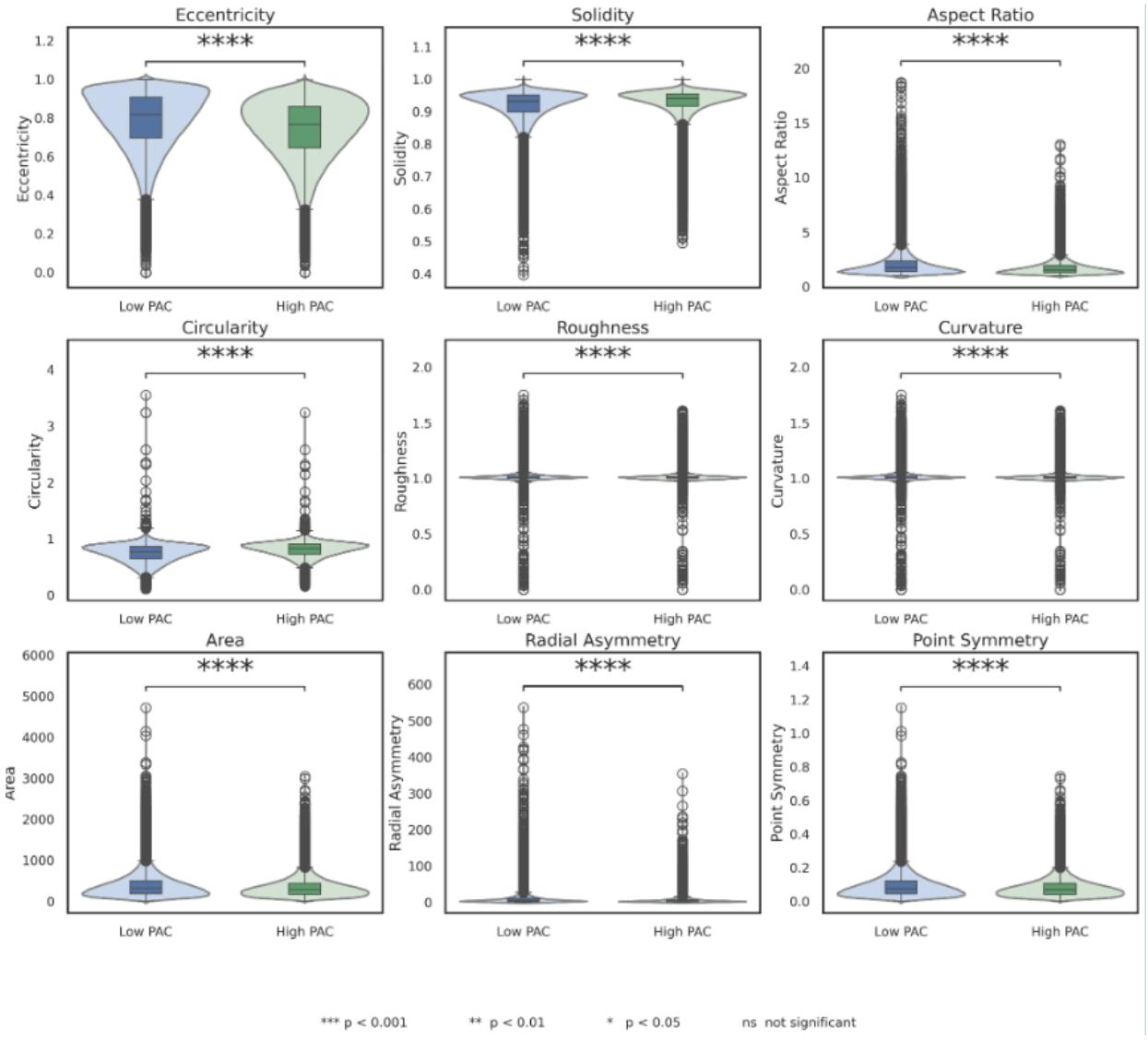


**Figure 2.9: Violin plots of the distributions of each morphometric feature.**

*In this, "Low PAC" refers to the 1st percentile PAC cohort and "High PAC" refers to the 99th percentile group. The legend for the results of the Mann-Whitney U test are below the violin plots.*

According to the median values, the typical nucleus in the 99th percentile (MSI-H) cohort was more compact and uniform than in the 1st percentile (MSS) cohort. Specifically, it had a 1.11x smaller median area and was significantly less elongated, with a 1.12x lower mean aspect ratio and 1.08x higher circularity (Table 2.5, Table 2.6). While these nuclei tended to be rounder, they still had features of asymmetry, evidenced by 1.11x lower median point symmetry (Table 2.5, Table 2.6).

On the other hand, the 1st percentile PAC nuclei were larger, more elongated and possessed more irregular boundaries, as shown by their lower median solidity and higher median radial asymmetry (Table 2.5).

**Table 2.5: Morphometric analysis of the average nuclei in the 1st percentile PAC cohort.**

| | **mean** | **median** |
|---|---|---|
| **Eccentricity** | 0.720 | 0.817 |
| **Solidity** | 0.921 | 0.931 |
| **Aspect Ratio** | 1.604 | 1.737 |
| **Cir cularity** | 0.80 | 0.772 |
| **Curvature** | 1.029 | 1.007 |
| **Area** | 366.255 | 319.000 |
| **Radial Asymmetry** | 5.170 | 4.906 |
| **Point Symmetry** | 0.0889 | 0.0769 |

**Table 2.6. Morphometric analysis of the average nuclei in the 99th percentile PAC cohort.**

| | mean | median |
|---|---|---|
| **Eccentricity** | 0.738 | 0.7656 |
| **Solidity** | 0.929 | 0.940 |
| **Aspect Ratio** | 1.734 | 1.555 |
| **Circularity** | 0.808 | 0.831 |
| **Curvature** | 1.014 | 1.005 |
| **Area** | 336.891 | 288.000 |
| **Radial Asymmetry** | 5.690 | 2.933 |
| **Point Symmetry** | 0.081 | 0.0696 |

From a biological perspective, these contrasting morphometric profiles could be a result of both tumor genomic and microenvironmental attributes. The overall nuclear profile in the 99th percentile PAC (MSI-H representative) patches (small, compact, uniform) could be a result of the high lymphocyte cell infiltration (Table 2.4), whose nuclei tend to be rounder and smaller [111]. On the other hand, the increased median asymmetry of the 1st percentile PAC (MSS) could be either due to the large percentage of stromal and connective associated cells, which tend to have a “stretched” appearance or distortions in nuclear structure and stability from Chromosomal Instability (CNI), which typically higher in MSS tumors with high loss of heterozygosity [112].

#### 2.6.1.4 Summary of Case Study 1 Results

Overall, our analysis showed that the weakly-supervised model successfully learned the correct underlying TME that differentiates MSI-H versus MSS colorectal cancers. PAC allowed us to identify the model's learned 'class phenotypes' by isolating the instances with the most extreme scores, which directly corresponded to the known biological differences between the two classes. For example, the model learned to define MSI-H by a highly immunogenic and disorganized TME, where tumor cells were more scattered across the stroma than in clusters like in the MSS tumors (Figure 2.7, 2.8). This ill-defined stroma-tumor boundary has been previously reported in the literature as a potential spatial biomarker for MSI-H [89].

While these findings largely validate the model's learning, the PAC analysis also exposed the basis for its performance limitations. For instance, the model strongly associated high levels of necrosis with its MSI-H phenotype, a feature that is typically found less in this molecular subtype [90]. This reliance on a potentially confounding feature, alongside the correct immunological signals, provides a potential rationale for why the model performed well at an AUC of 0.82, but was not perfect. This demonstrates PAC's utility in diagnosing a model's specific points of failure while uncovering biologically meaningful features relevant to model decisions.

### 2.6.2 Case Study 2: B-ALL

#### *2.6.2.1 Comparative Analysis of Instance Selection Metrics*

An analysis of the extreme cohorts selected by each metric demonstrated that each method prioritized different cellular features (Figure 2.10). For instance, when stratifying based on extreme probability scores, the non-B-ALL representative cohort (1st percentile PAC) was

primarily characterized by high median values of forward scatter (FSC) and side scatter (SSC), suggesting a focus on physical cell properties like size and granularity. The corresponding 99th percentile lacked a clearly defined immunophenotype. When attention scores were used for selection, the resulting 99th percentile cohort showed high median expression for known B-ALL markers such as CD10 and CD34, but the largest difference between the 99th and 1st percentiles was still the SSC and FSC values. A larger weight/attention was given to cells with a smaller FSC and SSC (representative of smaller blood cells like lymphocytes) that also had lower CD66b expression (a granulocyte cell marker).

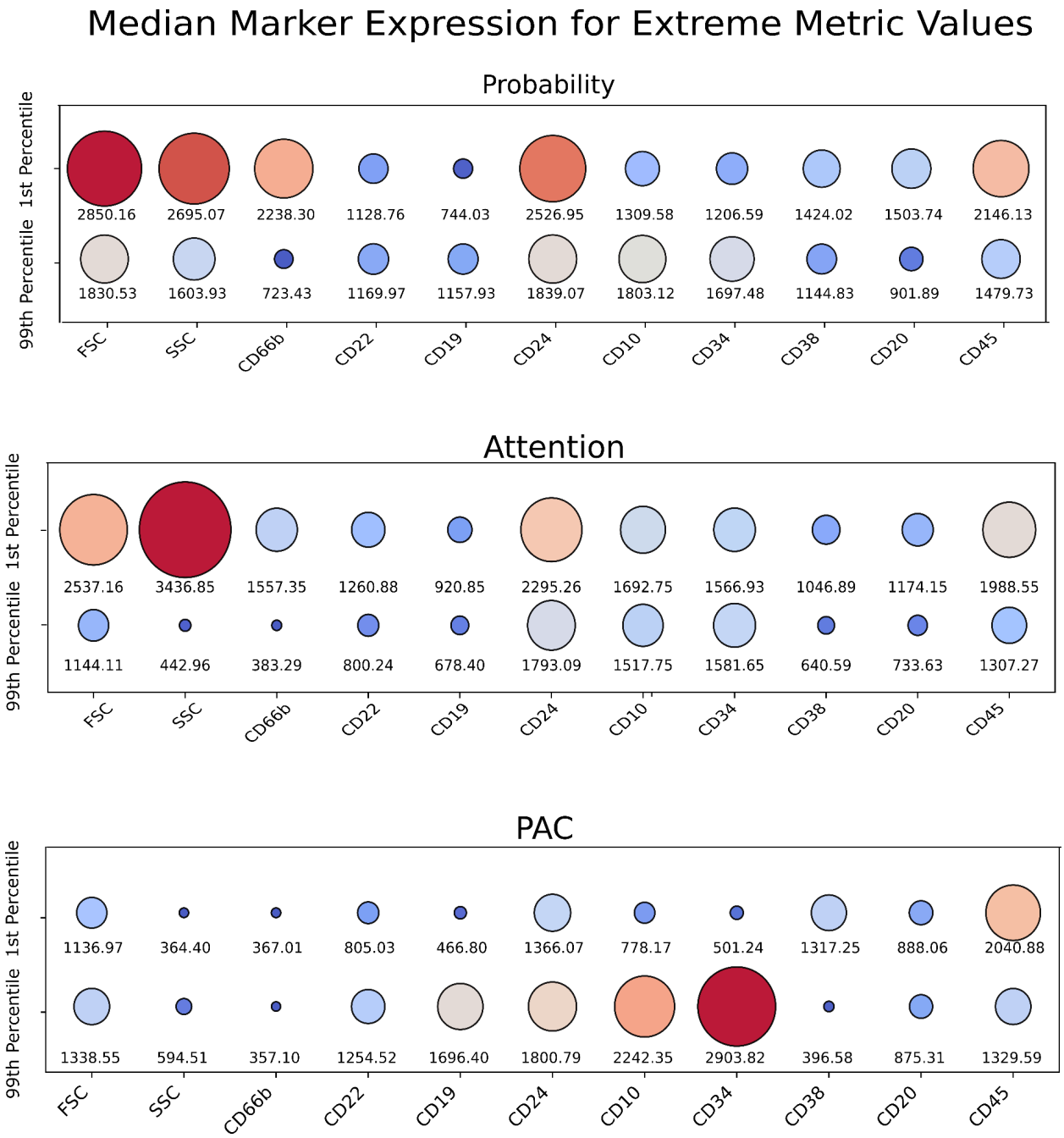


**Figure 2.10: Analysis of Extreme Cell cohorts Selected by Probability, Attention, and PAC**

*This figure showcases the differences in polarization of instances between the different interpretability metrics that are often used in the field. Each comparison includes the median values of expression for cells in the 1st and 99th percentiles in each metric. The first box represents probability, the middle attention, and the last PAC.*

The extreme cohorts obtained with PAC were quite different from using probability or attention alone. Both the 99th and 1st percentile cohorts exhibit low FSC and SSC scores,

accompanied by extremely low expression of the granulocyte marker CD66b. Based on these findings, it is clear that the model was defining both extremes as different lymphocyte lineages (Figure 2.10).

The 1st percentile PAC represents the model's definition of the non-B-ALL immunophenotype. This cohort was precisely characterized by a high median expression of the pan-leukocyte marker CD45 (2040.88), but notably low expression of the B-cell lineage marker CD19 (466.80) and the progenitor markers CD10 (778.17) and CD34 (501.24). This specific immunoprofile—being CD45 positive with low side scatter is most consistent with mature T-cells or NK cell populations. For example, outside of B-cells, CD24 is more often expressed in regulatory T-cells [113].

In stark contrast the model's definition of the B-ALL immunophenotype (99th percentile PAC) was unambiguously defined by a dominant expression of the canonical B-ALL blast markers CD10 (2242.35) and CD34 (1903.82). This blast phenotype was further supported by the concurrently high expression of key B-lineage markers, including CD24 (1800.79), CD19 (1696.40), and CD22 (1254.52). The analysis demonstrates that PAC successfully separated and defined the two opposing phenotypes: a classic CD10+/CD34+ B-ALL blast population versus a mature Treg-cell/NK-cell population that the model learned was the most definitive example of a non-B-ALL phenotype by contrasting benign leukocyte populations versus those of B-ALL. Comparing these results to those of attention or probability alone, PAC allows for a more truthful interpretation of the model's biological reasoning that is appropriate to its performance of 1.0 AUC.

## 2.8. Discussion

Across two distinct case studies utilizing different data modalities, the application of Polarized Attentional Certainty (PAC) consistently enabled the translation of abstract, weakly-supervised model decisions into quantifiable and biologically relevant insights. The significance of this work lies not only in the introduction of a novel interpretability metric but also in its demonstrated ability to validate, critique, and deeply understand the learning process of complex models in biomedicine.

In the computational pathology task, PAC demonstrated that the model had independently learned to classify MSI-H versus MSS colorectal cancer instances based on the immunological phenotypes that separate these two biomolecular subtypes (immune hot versus immune cold). By isolating the extreme instance cohorts, we confirmed that the model associated MSI-H with an inflamed tumor microenvironment rich in lymphocytes and necrosis, while associating MSS with a dense, non-immunogenic mixture of tumor cells and stroma. This in itself also revealed part of the model's limitations, its reliance on necrosis as a feature of MSI-H. Necrosis in itself is usually not associated with MSI-H, instead necrotic-like tissue in colorectal cancer is often associated with dirty necrosis, a feature common in MSS colorectal tumors. This showcases PAC ability to validate the model's biological reasoning and uncover any biases that may be impacting its generalizability.

In the flow cytometry case study, PAC proved superior to its counterparts (attention and probability alone) in terms of understanding the model's learned phenotype for both classes. This

showcases the issue of using attention or probability alone in weakly-supervised multi-instance learning models, where individual metrics can be ambiguous for interpretability given either their lack of directionality (attention) or insights into instance contributions (probability). By combining both and creating a polarized score, PAC successfully that a perfect-scoring model had learned to define B-ALL by the canonical CD10+/CD34+ blast immunophenotype, while defining the non-B-ALL class by a mature T-cell/NK-cell population. The model's ability to learn such clinically relevant decision boundaries demonstrates a level of nuanced understanding that would be difficult to deliver into the clinical setting without this clear, interpretable framework.

In both applications, PAC served as a robust metric for deep learning interpretation. It effectively bridged the gap between the model's internal logic and the specific, verifiable biological features it uses for classification, proving its utility for both validation of a model's successes and diagnosing its potential biases.

Despite its demonstrated utility, the PAC framework and its application in this thesis does have several important limitations. Due to the nature of computational pathology models, where instances are images instead of direct biological measurements like in FCM, the downstream analysis relies on a series of pre-trained, secondary models to extract these measurements. These models, while powerful and well established, have their own inherent margins of error that may affect the final quantification. Because of this, the quantitative compositional and morphometric data presented should be interpreted as strong, model-driven evidence rather than the absolute ground truth. Any classification or segmentation errors in these secondary models could act as a confounding variable in the final biological interpretation. Finally, the scope of this work is focused on specific attention-based MIL architectures for binary classification tasks. The direct

applicability of the current PAC formulation to other model architectures with attention backbones (e.g., visual transformers, transformers, attention graph neural networks, etc..) or other tasks like regression has not been explored.

These limitations open up this work for several directions for future research. Primarily future works should focus on the extension of PAC to multi-class classifications. This would likely require adapting the log-odds formulation by employing a one-vs-rest approach for each class, to generate a score for each potential outcome. An essential future work would also be to validate PAC's utility on a wider range of model architectures, including different attention mechanisms (e.g., linearized attention) and even transformer-based models to establish its generalizability.

The most significant future directions would be to investigate the prognostic utility of PAC scores. The current case studies were diagnostic, however, the density and characteristics of the extreme PAC cohorts from a well-performing model could correlate with patient outcomes, such as response to immunotherapy or overall survival (e.g., 90% of the model decisions had a PAC score of $> 0.5$, indicative of a very strong MSI-H case with high levels of immune cell infiltration). Furthermore, PAC could be integrated into a human-in-the-loop active learning framework, where pathologists can provide feedback to a model based on its identified PAC distribution. By identifying instances that are ambiguous (PAC scores near zero) or those that are clearly misclassified (high PAC scores for the wrong class), the system could flag these specific regions for pathologist review, obtain feedback, and iteratively refine the model.

## 2.7. Conclusion

The increasing integration of deep learning into biomedical research has created a critical need for methods that can remove the barriers of "black-box" reasoning. This inability to understand the predictive reasoning of these models is a significant barrier to their clinical translation and scientific utility. This thesis chapter addressed this challenge by developing and validating a new metric called Polarized Attentional Certainty (PAC). PAC was specifically designed to provide a more robust and coherent understanding of weakly supervised, attention-based models by integrating the model's predictive certainty and its focus. By doing so, PAC provides a unified score that reflects how a model is defining the differences between positive and negative classes.

The primary contribution of this work is the establishment of PAC as a comprehensive, flexible framework to study model thought and analyze its potential learned biases. The metric was tested in two very different data modalities within the biomodalities and showed that how these model decisions due lie in some form of biological relevance. At the same time, it showed that models, like in the MSI-H versus MSS classification, can learn spurious correlations that impact their generalizability.

In conclusion, this thesis demonstrates that by moving beyond dissociated metrics of probability and attention, we can achieve a deeper and more actionable understanding of complex deep learning models. Tools like PAC represent a critical step forward in transforming these models from opaque predictors into transparent partners in scientific discovery. As artificial intelligence continues to evolve, such interpretability frameworks will be indispensable

for ensuring that these powerful technologies are not only accurate but also insightful, reliable, and ultimately, trustworthy.

## 2.8. Supplemental Material

### 2.8.1 Data and Code Availability

- The jupyter notebooks used for the PAC project can be found at https://github.com/lolmomarchal/PAC .
- The WSI from the TCGA cohort are available from https://portal.gdc.cancer.gov/.
- MSI status for the TCGA-COAD cohort is available at https://github.com/KatherLab/cancer-metadata/blob/main/tcga/liu.xlsx.
- The Mutographs dataset is not publicly available.
- The B-ALL dataset and the proportion labels can be found at https://github.com/erobl/csnn.
- All additional models can be accessed through the TiaToolBox documentation https://tia-toolbox.readthedocs.io/.

REFERENCES

[1]

M. Mubarak, “Move from traditional histopathology to digital and computational pathology: Are we ready?,” *Indian J Nephrol*, vol. 32, no. 5, p. 414, 2022, doi: 10.4103/ijn.IJN_508_20.

[2]

M. Cui and D. Y. Zhang, “Artificial intelligence and computational pathology,” *Lab Invest*, vol. 101, no. 4, pp. 412–422, Apr. 2021, doi: 10.1038/s41374-020-00514-0.

[3]

T. T. Rau, W. Cross, R. R. Lastra, R. C. Lo, A. Matoso, and C. S. Herrington, “Closing the loop – the role of pathologists in digital and computational pathology research,” *J Pathol Clin Res*, vol. 10, no. 2, p. e12366, Mar. 2024, doi: 10.1002/2056-4538.12366.

[4]

J. Levy, C. Haudenschild, C. Barwick, B. Christensen, and L. Vaickus, “Topological Feature Extraction and Visualization of Whole Slide Images using Graph Neural Networks,” *Pac Symp Biocomput*, vol. 26, pp. 285–296, 2021, Accessed: Sep. 11, 2025. [Online]. Available: https://www.ncbi.nlm.nih.gov/pmc/articles/PMC7959046/

[5]

N. Dimitriou, O. Arandjelović, and P. D. Caie, “Deep Learning for Whole Slide Image Analysis: An Overview,” *Front Med (Lausanne)*, vol. 6, p. 264, Nov. 2019, doi: 10.3389/fmed.2019.00264.

[6]

M. Poddar, J. S. Marwaha, W. Yuan, S. Romero-Brufau, and G. A. Brat, “An operational guide to translational clinical machine learning in academic medical centers,” *npj Digit. Med.*, vol. 7, no. 1, pp. 1–7, May 2024, doi: 10.1038/s41746-024-01094-9.

[7]

M. Ennab and H. Mcheick, “Enhancing interpretability and accuracy of AI models in healthcare: a comprehensive review on challenges and future directions,” *Front Robot AI*, vol. 11, p. 1444763, Nov. 2024, doi: 10.3389/frobt.2024.1444763.

[8]

N. Kanwal, F. Pérez-Bueno, A. Schmidt, K. Engan, and R. Molina, “The Devil is in the Details: Whole Slide Image Acquisition and Processing for Artifacts Detection, Color

Variation, and Data Augmentation: A Review," *IEEE Access*, vol. 10, pp. 58821–58844, 2022, doi: 10.1109/ACCESS.2022.3176091.

[9]

Md. S. Hossain *et al.*, "Tissue Artifact Segmentation and Severity Assessment for Automatic Analysis Using WSI," *IEEE Access*, vol. 11, pp. 21977–21991, 2023, doi: 10.1109/ACCESS.2023.3250556.

[10]

E. N. Bergstrom *et al.*, "Deep Learning Artificial Intelligence Predicts Homologous Recombination Deficiency and Platinum Response From Histologic Slides," *JCO*, vol. 42, no. 30, pp. 3550–3560, Oct. 2024, doi: 10.1200/JCO.23.02641.

[11]

S. C. Wetstein *et al.*, "Deep learning-based breast cancer grading and survival analysis on whole-slide histopathology images," *Sci Rep*, vol. 12, no. 1, p. 15102, Sep. 2022, doi: 10.1038/s41598-022-19112-9.

[12]

J. N. Kather *et al.*, "Deep learning can predict microsatellite instability directly from histology in gastrointestinal cancer," *Nat Med*, vol. 25, no. 7, pp. 1054–1056, Jul. 2019, doi: 10.1038/s41591-019-0462-y.

[13]

Y. Zheng, C. Sadée, M. Ozawa, B. E. Howitt, and O. Gevaert, "Single-cell multimodal analysis reveals tumor microenvironment predictive of treatment response in non-small cell lung cancer," *Sci Adv*, vol. 11, no. 21, p. eadu2151, May 2025, doi: 10.1126/sciadv.adu2151.

[14]

S. Graham *et al.*, "Hover-Net: Simultaneous segmentation and classification of nuclei in multi-tissue histology images," *Medical Image Analysis*, vol. 58, p. 101563, Dec. 2019, doi: 10.1016/j.media.2019.101563.

[15]

M. Y. Lu, D. F. K. Williamson, T. Y. Chen, R. J. Chen, M. Barbieri, and F. Mahmood, "Data-efficient and weakly supervised computational pathology on whole-slide images," *Nat Biomed Eng*, vol. 5, no. 6, pp. 555–570, Jun. 2021, doi: 10.1038/s41551-020-00682-w.

[16]

Y. Zheng *et al.*, “A Graph-Transformer for Whole Slide Image Classification,” *IEEE Trans Med Imaging*, vol. 41, no. 11, pp. 3003–3015, Nov. 2022, doi: 10.1109/TMI.2022.3176598.

[17]

A. Asif, K. Rajpoot, S. Graham, D. Snead, F. Minhas, and N. Rajpoot, “Unleashing the potential of AI for pathology: challenges and recommendations,” *J Pathol*, vol. 260, no. 5, pp. 564–577, Aug. 2023, doi: 10.1002/path.6168.

[18]

J. E. M. Swillens, I. D. Nagtegaal, S. Engels, A. Lugli, R. P. M. G. Hermens, and J. A. W. M. Van Der Laak, “Pathologists’ first opinions on barriers and facilitators of computational pathology adoption in oncological pathology: an international study,” *Oncogene*, vol. 42, no. 38, pp. 2816–2827, Sep. 2023, doi: 10.1038/s41388-023-02797-1.

[19]

Q. Huang, S. Wu, Z. Ou, and Y. Gao, “Computational pathology: A comprehensive review of recent developments in digital and intelligent pathology,” *Intelligent Oncology*, vol. 1, no. 2, pp. 139–159, Apr. 2025, doi: 10.1016/j.intonc.2025.03.004.

[20]

M. A. Berbís *et al.*, “Computational pathology in 2030: a Delphi study forecasting the role of AI in pathology within the next decade,” *EBioMedicine*, vol. 88, p. 104427, Feb. 2023, doi: 10.1016/j.ebiom.2022.104427.

[21]

S. J. Wagner *et al.*, “Built to Last? Reproducibility and Reusability of Deep Learning Algorithms in Computational Pathology,” *Modern Pathology*, vol. 37, no. 1, p. 100350, Jan. 2024, doi: 10.1016/j.modpat.2023.100350.

[22]

M. Afonso, P. M. S. Bhawsar, M. Saha, J. S. Almeida, and A. L. Oliveira, “Multiple Instance Learning for WSI: A comparative analysis of attention-based approaches,” *J Pathol Inform*, vol. 15, p. 100403, Oct. 2024, doi: 10.1016/j.jpi.2024.100403.

[23]

K. Ashman *et al.*, “Whole slide image data utilization informed by digital diagnosis patterns,” *Journal of Pathology Informatics*, vol. 13, p. 100113, 2022, doi: 10.1016/j.jpi.2022.100113.

[24]

F. Aeffner *et al.*, “Introduction to Digital Image Analysis in Whole-slide Imaging: A White Paper from the Digital Pathology Association,” *Journal of Pathology Informatics*, vol. 10, no. 1, p. 9, Jan. 2019, doi: 10.4103/jpi.jpi_82_18.

[25]

A. Patil *et al.*, “Efficient quality control of whole slide pathology images with human-in-the-loop training,” *Journal of Pathology Informatics*, vol. 14, p. 100306, 2023, doi: 10.1016/j.jpi.2023.100306.

[26]

Z. Weng *et al.*, “GrandQC: A comprehensive solution to quality control problem in digital pathology,” *Nat Commun*, vol. 15, no. 1, p. 10685, Dec. 2024, doi: 10.1038/s41467-024-54769-y.

[27]

A. Jurgas, M. Wodzinski, M. D’Amato, J. Van Der Laak, M. Atzori, and H. Müller, “Improving quality control of whole slide images by explicit artifact augmentation,” *Sci Rep*, vol. 14, no. 1, p. 17847, Aug. 2024, doi: 10.1038/s41598-024-68667-2.

[28]

K. Falahkheirkhah *et al.*, “A generative adversarial approach to facilitate archival-quality histopathologic diagnoses from frozen tissue sections,” *Laboratory Investigation*, vol. 102, no. 5, pp. 554–559, May 2022, doi: 10.1038/s41374-021-00718-y.

[29]

Y. Zhang, Z. Gao, K. He, C. Li, and R. Mao, “From patches to WSIs: A systematic review of deep Multiple Instance Learning in computational pathology,” *Information Fusion*, vol. 119, p. 103027, Jul. 2025, doi: 10.1016/j.inffus.2025.103027.

[30]

N. Otsu, “A Threshold Selection Method from Gray-Level Histograms,” *IEEE Trans. Syst., Man, Cybern.*, vol. 9, no. 1, pp. 62–66, Jan. 1979, doi: 10.1109/TSMC.1979.4310076.

[31]

B. A. Schreiber *et al.*, “Rapid artefact removal and H&E-stained tissue segmentation,” *Sci Rep*, vol. 14, no. 1, p. 309, Jan. 2024, doi: 10.1038/s41598-023-50183-4.

[32]

P. J. Tadrous, “On the concept of objectivity in digital image analysis in pathology,” *Pathology*, vol. 42, no. 3, pp. 207–211, Apr. 2010, doi: 10.3109/00313021003641758.

[33]

P. Arbeláez, M. Maire, C. Fowlkes, and J. Malik, “Contour Detection and Hierarchical Image Segmentation,” *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 33, no. 5, pp. 898–916, May 2011, doi: 10.1109/TPAMI.2010.161.

[34]

J. Canny, “A Computational Approach to Edge Detection,” *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. PAMI-8, no. 6, pp. 679–698, Nov. 1986, doi: 10.1109/TPAMI.1986.4767851.

[35]

O. S. M. El Nahhas *et al.*, “From whole-slide image to biomarker prediction: end-to-end weakly supervised deep learning in computational pathology,” *Nat Protoc*, vol. 20, no. 1, pp. 293–316, Jan. 2025, doi: 10.1038/s41596-024-01047-2.

[36]

A. Marcolini, N. Bussola, E. Arbitrio, M. Amgad, G. Jurman, and C. Furlanello, “histolab: A Python library for reproducible Digital Pathology preprocessing with automated testing,” *SoftwareX*, vol. 20, p. 101237, Dec. 2022, doi: 10.1016/j.softx.2022.101237.

[37]

S. Kothari, J. H. Phan, and M. D. Wang, “Eliminating tissue-fold artifacts in histopathological whole-slide images for improved image-based prediction of cancer grade,” *Journal of Pathology Informatics*, vol. 4, no. 1, p. 22, Jan. 2013, doi: 10.4103/2153-3539.117448.

[38]

S. R, J. R, and S. M. A, “Computer vision approach for motion blur image restoration system,” in *2023 14th International Conference on Computing Communication and Networking Technologies (ICCCNT)*, Jul. 2023, pp. 1–9. doi: 10.1109/ICCCNT56998.2023.10306622.

[39]

P. Chilukuri, S. Harshitha, P. Abhiram, S. Susanna, S. Vyshnavi, and B. C. Babu, “Analysing Of Image Quality Computation Models Through Convolutional Neural Network,” in *2023 2nd International Conference on Vision Towards Emerging Trends in Communication and Networking Technologies (ViTECoN)*, May 2023, pp. 1–7. doi: 10.1109/ViTECoN58111.2023.10157161.

[40]

J. R. Kaczmarzyk, J. H. Saltz, and P. K. Koo, “Explainable AI for computational pathology identifies model limitations and tissue biomarkers,” *ArXiv*, p.

arXiv:2409.03080v2, Nov. 2024, Accessed: Sep. 11, 2025. [Online]. Available: https://www.ncbi.nlm.nih.gov/pmc/articles/PMC11398542/

[41]

E. Reinhard, M. Adhikhmin, B. Gooch, and P. Shirley, “Color transfer between images,” *IEEE Comput. Grap. Appl.*, vol. 21, no. 4, pp. 34–41, Aug. 2001, doi: 10.1109/38.946629.

[42]

M. Macenko *et al.*, “A method for normalizing histology slides for quantitative analysis,” in *2009 IEEE International Symposium on Biomedical Imaging: From Nano to Macro*, Jun. 2009, pp. 1107–1110. doi: 10.1109/ISBI.2009.5193250.

[43]

A. Vahadane *et al.*, “Structure-Preserving Color Normalization and Sparse Stain Separation for Histological Images,” *IEEE Trans. Med. Imaging*, vol. 35, no. 8, pp. 1962–1971, Aug. 2016, doi: 10.1109/TMI.2016.2529665.

[44]

H. Kang *et al.*, “StainNet: A Fast and Robust Stain Normalization Network,” *Front. Med.*, vol. 8, p. 746307, Nov. 2021, doi: 10.3389/fmed.2021.746307.

[45]

M. J. Hetz, T.-C. Bucher, and T. J. Brinker, “Multi-domain stain normalization for digital pathology: A cycle-consistent adversarial network for whole slide images,” *Medical Image Analysis*, vol. 94, p. 103149, May 2024, doi: 10.1016/j.media.2024.103149.

[46]

M. T. Shaban, C. Baur, N. Navab, and S. Albarqouni, “StainGAN: Stain Style Transfer for Digital Histological Images,” 2018, *arXiv*. doi: 10.48550/ARXIV.1804.01601.

[47]

S. Roy, S. Panda, and M. Jangid, “Modified Reinhard Algorithm for Color Normalization of Colorectal Cancer Histopathology Images,” in *2021 29th European Signal Processing Conference (EUSIPCO)*, Dublin, Ireland: IEEE, Aug. 2021, pp. 1231–1235. doi: 10.23919/EUSIPCO54536.2021.9616117.

[48]

K. He, X. Zhang, S. Ren, and J. Sun, “Deep Residual Learning for Image Recognition,” Dec. 10, 2015, *arXiv*: arXiv:1512.03385. doi: 10.48550/arXiv.1512.03385.

[49]

M. Y. Lu *et al.*, “A visual-language foundation model for computational pathology,” *Nat Med*, vol. 30, no. 3, pp. 863–874, Mar. 2024, doi: 10.1038/s41591-024-02856-4.

[50]

E. Vorontsov *et al.*, “A foundation model for clinical-grade computational pathology and rare cancers detection,” *Nat Med*, vol. 30, no. 10, pp. 2924–2935, Oct. 2024, doi: 10.1038/s41591-024-03141-0.

[51]

J. M. Dolezal *et al.*, “Slideflow: deep learning for digital histopathology with real-time whole-slide visualization,” *BMC Bioinformatics*, vol. 25, no. 1, p. 134, Mar. 2024, doi: 10.1186/s12859-024-05758-x.

[52]

J. Rosenthal *et al.*, “Building Tools for Machine Learning and Artificial Intelligence in Cancer Research: Best Practices and a Case Study with the PathML Toolkit for Computational Pathology,” *Molecular Cancer Research*, vol. 20, no. 2, pp. 202–206, Feb. 2022, doi: 10.1158/1541-7786.MCR-21-0665.

[53]

J. Pocock *et al.*, “TIAToolbox as an end-to-end library for advanced tissue image analytics,” *Commun Med*, vol. 2, no. 1, pp. 1–14, Sep. 2022, doi: 10.1038/s43856-022-00186-5.

[54]

R. J. Chen *et al.*, “Towards a general-purpose foundation model for computational pathology,” *Nat Med*, vol. 30, no. 3, pp. 850–862, Mar. 2024, doi: 10.1038/s41591-024-02857-3.

[55]

K. Faryna, J. van der Laak, and G. Litjens, “Automatic data augmentation to improve generalization of deep learning in H&E stained histopathology,” *Computers in Biology and Medicine*, vol. 170, p. 108018, Mar. 2024, doi: 10.1016/j.compbiomed.2024.108018.

[56]

A. Partin *et al.*, “Data augmentation and multimodal learning for predicting drug response in patient-derived xenografts from gene expressions and histology images,” *Front. Med.*, vol. 10, Mar. 2023, doi: 10.3389/fmed.2023.1058919.

[57]

N. Kanwal *et al.*, “Equipping computational pathology systems with artifact processing pipelines: a showcase for computation and performance trade-offs,” *BMC Med Inform Decis Mak*, vol. 24, no. 1, p. 288, Oct. 2024, doi: 10.1186/s12911-024-02676-z.

[58]

A. S. Kanse *et al.*, “Cautious Artificial Intelligence Improves Outcomes and Trust by Flagging Outlier Cases,” *JCO Clinical Cancer Informatics*, no. 6, p. e2200067, Oct. 2022, doi: 10.1200/CCI.22.00067.

[59]

C. A. Barbano and A. Pedersen, *EIDOSLAB/torchstain: v1.2.0-stable*. (Aug. 10, 2022). Zenodo. doi: 10.5281/ZENODO.6979540.

[60]

“Colorectal Cancer Statistics | How Common Is Colorectal Cancer?” Accessed: Sep. 11, 2025. [Online]. Available: https://www.cancer.org/cancer/types/colon-rectal-cancer/about/key-statistics.html

[61]

C. R. Boland and A. Goel, “Microsatellite Instability in Colorectal Cancer,” *Gastroenterology*, vol. 138, no. 6, pp. 2073-2087.e3, Jun. 2010, doi: 10.1053/j.gastro.2009.12.064.

[62]

F. Battaglin, M. Naseem, H.-J. Lenz, and M. E. Salem, “Microsatellite Instability in Colorectal Cancer: Overview of Its Clinical Significance and Novel Perspectives,” *Clin Adv Hematol Oncol*, vol. 16, no. 11, pp. 735–745, Nov. 2018, Accessed: Sep. 11, 2025. [Online]. Available: https://www.ncbi.nlm.nih.gov/pmc/articles/PMC7493692/

[63]

V. Koch *et al.*, “DinoBloom: A Foundation Model for Generalizable Cell Embeddings in Hematology,” Apr. 07, 2024, *arXiv*: arXiv:2404.05022. doi: 10.48550/arXiv.2404.05022.

[64]

“STAR: A Fast and Robust Rigid Registration Framework for Serial Histopathological Images.” Accessed: Sep. 11, 2025. [Online]. Available: https://arxiv.org/html/2509.02952v1

[65]

C. for D. and R. Health, “Good Machine Learning Practice for Medical Device Development: Guiding Principles,” *FDA*, Mar. 2025, Accessed: Sep. 11, 2025. [Online]. Available:

https://www.fda.gov/medical-devices/software-medical-device-samd/good-machine-learning-practice-medical-device-development-guiding-principles

[66]

V. Muralidharan *et al.*, “A scoping review of reporting gaps in FDA-approved AI medical devices,” *npj Digit. Med.*, vol. 7, no. 1, p. 273, Oct. 2024, doi: 10.1038/s41746-024-01270-x.

[67]

G. Verghese *et al.*, “Computational pathology in cancer diagnosis, prognosis, and prediction – present day and prospects,” *The Journal of Pathology*, vol. 260, no. 5, pp. 551–563, Aug. 2023, doi: 10.1002/path.6163.

[68]

C. for D. and R. Health, “De Novo Classification Request,” *FDA*, Jul. 2025, Accessed: Sep. 11, 2025. [Online]. Available: https://www.fda.gov/medical-devices/premarket-submissions-selecting-and-preparing-correct-submission/de-novo-classification-request

[69]

A. Vaidya *et al.*, “Demographic bias in misdiagnosis by computational pathology models,” *Nat Med*, vol. 30, no. 4, pp. 1174–1190, Apr. 2024, doi: 10.1038/s41591-024-02885-z.

[70]

CDC, “Cancer and African American People,” Cancer. Accessed: Sep. 11, 2025. [Online]. Available: https://www.cdc.gov/cancer/health-equity/african-american.html

[71]

L. Huang, Z. Chen, Z. Yang, and W. Huang, “Advancing Healthcare Accessibility: Fusing Artificial Intelligence with Flexible Sensing to Forge Digital Health Innovations,” *BME Front*, vol. 5, p. 0062, doi: 10.34133/bmef.0062.

[72]

K. Simonyan, A. Vedaldi, and A. Zisserman, “Deep Inside Convolutional Networks: Visualising Image Classification Models and Saliency Maps,” Apr. 19, 2014, *arXiv*: arXiv:1312.6034. doi: 10.48550/arXiv.1312.6034.

[73]

S. Lundberg and S.-I. Lee, “A Unified Approach to Interpreting Model Predictions,” Nov. 25, 2017, *arXiv*: arXiv:1705.07874. doi: 10.48550/arXiv.1705.07874.

[74]

M. T. Ribeiro, S. Singh, and C. Guestrin, "'Why Should I Trust You?': Explaining the Predictions of Any Classifier," Aug. 09, 2016, *arXiv*: arXiv:1602.04938. doi: 10.48550/arXiv.1602.04938.

[75]

J. Zhang, H. Chao, G. Dasegowda, G. Wang, M. K. Kalra, and P. Yan, "Revisiting the Trustworthiness of Saliency Methods in Radiology AI," *Radiol Artif Intell*, vol. 6, no. 1, p. e220221, Nov. 2023, doi: 10.1148/ryai.220221.

[76]

N. Arun *et al.*, "Assessing the Trustworthiness of Saliency Maps for Localizing Abnormalities in Medical Imaging," *Radiology: Artificial Intelligence*, vol. 3, no. 6, p. e200267, Nov. 2021, doi: 10.1148/ryai.2021200267.

[77]

A. Vaswani *et al.*, "Attention Is All You Need," Aug. 02, 2023, *arXiv*: arXiv:1706.03762. doi: 10.48550/arXiv.1706.03762.

[78]

W. Bonnaffé *et al.*, "Beyond attention: deriving biologically interpretable insights from weakly-supervised multiple-instance learning models," Sep. 07, 2023, *arXiv*: arXiv:2309.03925. doi: 10.48550/arXiv.2309.03925.

[79]

M. L. Nádorvári, G. Lotz, J. Kulka, A. Kiss, and J. Tímár, "Microsatellite instability and mismatch repair protein deficiency: equal predictive markers?," *Pathol Oncol Res*, vol. 30, p. 1611719, Apr. 2024, doi: 10.3389/pore.2024.1611719.

[80]

"https://www.cancer.gov/publications/dictionaries/cancer-terms/def/mismatch-repair-deficiency." Accessed: Sep. 11, 2025. [Online]. Available: https://www.cancer.gov/publications/dictionaries/cancer-terms/def/mismatch-repair-deficiency

[81]

M. C. Olave and R. P. Graham, "Mismatch repair deficiency: The what, how and why it is important," *Genes Chromosomes Cancer*, vol. 61, no. 6, pp. 314–321, Jun. 2022, doi: 10.1002/gcc.23015.

[82]

K. Venetis *et al.*, "Mismatch repair (MMR) and microsatellite instability (MSI) phenotypes across solid tumors: A comprehensive cBioPortal study on prevalence and

prognostic impact," *European Journal of Cancer*, vol. 217, p. 115233, Feb. 2025, doi: 10.1016/j.ejca.2025.115233.

[83]

C. Enrico Bena, J. Ollion, M. De Paepe, M. Ventroux, L. Robert, and M. Elez, "Real-time monitoring of replication errors' fate reveals the origin and dynamics of spontaneous mutations," *Nat Commun*, vol. 15, p. 2702, Mar. 2024, doi: 10.1038/s41467-024-46950-0.

[84]

A. M. Goodman, E. S. Sokol, G. M. Frampton, S. M. Lippman, and R. Kurzrock, "Microsatellite-Stable Tumors with High Mutational Burden Benefit from Immunotherapy," *Cancer Immunol Res*, vol. 7, no. 10, pp. 1570–1573, Oct. 2019, doi: 10.1158/2326-6066.CIR-19-0149.

[85]

Y. Li *et al.*, "Tumor Mutational Burden Predicting the Efficacy of Immune Checkpoint Inhibitors in Colorectal Cancer: A Systematic Review and Meta-Analysis," *Front Immunol*, vol. 12, p. 751407, Sep. 2021, doi: 10.3389/fimmu.2021.751407.

[86]

S. Narayanan *et al.*, "Tumor Infiltrating Lymphocytes and Macrophages Improve Survival in Microsatellite Unstable Colorectal Cancer," *Sci Rep*, vol. 9, no. 1, p. 13455, Sep. 2019, doi: 10.1038/s41598-019-49878-4.

[87]

E. Vargas‐Castellanos and A. Rincón‐Riveros, "Microsatellite Instability in the Tumor Microenvironment: The Role of Inflammation and the Microbiome," *Cancer Med*, vol. 14, no. 8, p. e70603, Apr. 2025, doi: 10.1002/cam4.70603.

[88]

Q. Wang, M. Yu, and S. Zhang, "The characteristics of the tumor immune microenvironment in colorectal cancer with different MSI status and current therapeutic strategies," *Front Immunol*, vol. 15, p. 1440830, Jan. 2025, doi: 10.3389/fimmu.2024.1440830.

[89]

Y. Feng *et al.*, "Spatially organized tumor-stroma boundary determines the efficacy of immunotherapy in colorectal cancer patients," *Nat Commun*, vol. 15, no. 1, p. 10259, Nov. 2024, doi: 10.1038/s41467-024-54710-3.

[90]

J.-F. Gao, G. Arbman, T. I. Wadhra, H. Zhang, and X.-F. Sun, “Relationships of tumor inflammatory infiltration and necrosis with microsatellite instability in colorectal cancers,” *World J Gastroenterol*, vol. 11, no. 14, pp. 2179–2183, Apr. 2005, doi: 10.3748/wjg.v11.i14.2179.

[91]

W.-S. Jo and J. M. Carethers, “Chemotherapeutic implications in microsatellite unstable colorectal cancer,” *Cancer Biomark*, vol. 2, no. 1–2, pp. 51–60, 2006, doi: 10.3233/cbm-2006-21-206.

[92]

F. Zhao *et al.*, “Predicting the Efficacy of 5-Fluorouracil-Based Adjuvant Chemotherapy in Gastric Cancer by Microsatellite Instability: A Meta-Analysis,” *J Environ Pathol Toxicol Oncol*, vol. 38, no. 1, pp. 21–28, 2019, doi: 10.1615/JEnvironPatholToxicolOncol.2018026876.

[93]

W. H. Gmeiner and C. C. Okechukwu, “Review of 5-FU resistance mechanisms in colorectal cancer: clinical significance of attenuated on-target effects,” *Cancer Drug Resist*, vol. 6, no. 2, pp. 257–272, Apr. 2023, doi: 10.20517/cdr.2022.136.

[94]

Y. Puckett and O. Chan, “Acute Lymphocytic Leukemia,” in *StatPearls*, Treasure Island (FL): StatPearls Publishing, 2025. Accessed: Sep. 11, 2025. [Online]. Available: http://www.ncbi.nlm.nih.gov/books/NBK459149/

[95]

Y. Wang, J. Liu, P. D. Burrows, and J.-Y. Wang, “B Cell Development and Maturation,” *Adv Exp Med Biol*, vol. 1254, pp. 1–22, 2020, doi: 10.1007/978-981-15-3532-1_1.

[96]

C. Schmitt, C. J. Eaves, and P. M. Lansdorp, “Expression of CD34 on human B cell precursors,” *Clin Exp Immunol*, vol. 85, no. 1, pp. 168–173, Jul. 1991, doi: 10.1111/j.1365-2249.1991.tb05699.x.

[97]

J. Y. Spiegel *et al.*, “CAR T cells with dual targeting of CD19 and CD22 in adult patients with recurrent or refractory B cell malignancies: a phase 1 trial,” *Nat Med*, vol. 27, no. 8, pp. 1419–1431, Aug. 2021, doi: 10.1038/s41591-021-01436-0.

[98]

H. Thorsson *et al.*, “Single-cell genomics details the maturation block in BCP-ALL and identifies therapeutic vulnerabilities in DUX4-r cases,” *Blood*, vol. 144, no. 13, pp. 1399–1411, Sep. 2024, doi: 10.1182/blood.2023021705.

[99]

R. K. Venugopalan *et al.*, “Leukemia-associated aberrant immunophenotype: A flow cytometry-based experience of 110 cases from a tertiary care center in Northern India,” *J Cancer Res Ther*, vol. 19, no. 5, pp. 1335–1339, 2023, doi: 10.4103/jcrt.jcrt_809_21.

[100]

J. Chen *et al.*, “Automated cytometric gating with human-level performance using bivariate segmentation,” *Nat Commun*, vol. 16, no. 1, p. 1576, Feb. 2025, doi: 10.1038/s41467-025-56622-2.

[101]

C. P. Verschoor, A. Lelic, J. L. Bramson, and D. M. E. Bowdish, “An Introduction to Automated Flow Cytometry Gating Tools and Their Implementation,” *Front. Immunol.*, vol. 6, Jul. 2015, doi: 10.3389/fimmu.2015.00380.

[102]

“Acute Lymphoblastic Leukemia Treatment - NCI.” Accessed: Sep. 11, 2025. [Online]. Available: https://www.cancer.gov/types/leukemia/patient/adult-all-treatment-pdq

[103]

J. E. Lewis, L. A. D. Cooper, D. L. Jaye, and O. Pozdnyakova, “Automated Deep Learning-Based Diagnosis and Molecular Characterization of Acute Myeloid Leukemia Using Flow Cytometry,” *Modern Pathology*, vol. 37, no. 1, p. 100373, Jan. 2024, doi: 10.1016/j.modpat.2023.100373.

[104]

Z. Ding and A. Baras, “Application and Characterization of the Multiple Instance Learning Framework in Flow Cytometry,” Jun. 12, 2025. doi: 10.1101/2025.06.10.658646.

[105]

H. Ma, J. Chen, J. T. Zhou, G. Wang, and C. Zhang, “Estimating LLM Uncertainty with Evidence,” May 09, 2025, *arXiv*: arXiv:2502.00290. doi: 10.48550/arXiv.2502.00290.

[106]

C. Tassi, J. Cedrique , A. Fitri, and R. Triebel, *The impact of averaging logits over probabilities on ensembles of neural networks*. The IJCAI-ECAI-22 Workshop on Artificial Intelligence Safety, 2022. [Online]. Available: https://ceur-ws.org/Vol-3215/19.pdf

[107]
L.-C. Chen, G. Papandreou, I. Kokkinos, K. Murphy, and A. L. Yuille, “DeepLab: Semantic Image Segmentation with Deep Convolutional Nets, Atrous Convolution, and Fully Connected CRFs,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 40, no. 4, pp. 834–848, Apr. 2018, doi: 10.1109/TPAMI.2017.2699184.

[108]
J. Gamper *et al.*, “PanNuke Dataset Extension, Insights and Baselines,” Apr. 22, 2020, *arXiv*: arXiv:2003.10778. doi: 10.48550/arXiv.2003.10778.

[109]
R. Verma *et al.*, “MoNuSAC2020: A Multi-Organ Nuclei Segmentation and Classification Challenge,” *IEEE Transactions on Medical Imaging*, vol. 40, no. 12, pp. 3413–3423, Dec. 2021, doi: 10.1109/TMI.2021.3085712.

[110]
R. van der Linde *et al.*, “Measurable Residual Disease (MRD) by Flow Cytometry in Adult B-Acute Lymphoblastic Leukaemia (B-ALL) and Acute Myeloid Leukaemia (AML): Correlation with Molecular MRD Testing and Clinical Outcome at One Year,” *Cancers (Basel)*, vol. 15, no. 20, p. 5064, Oct. 2023, doi: 10.3390/cancers15205064.

[111]
“Neutrophil and Lymphocytes.” Accessed: Sep. 11, 2025. [Online]. Available: https://imagebank.hematology.org/image/63363/neutrophil-and-lymphocytes?type=atlas

[112]
T. Sugai *et al.*, “Analysis of Molecular Alterations in Left- and Right-Sided Colorectal Carcinomas Reveals Distinct Pathways of Carcinogenesis,” *J Mol Diagn*, vol. 8, no. 2, pp. 193–201, May 2006, doi: 10.2353/jmoldx.2006.050052.

[113]
Y. Shi, J. Zhu, J.-Q. Liu, F. Talebian, M. Li, and X.-F. Bai, “CD24 is expressed on FoxP3+ regulatory T cells and regulates their function,” *Am J Transl Res*, vol. 14, no. 4, pp. 2291–2300, Apr. 2022, Accessed: Sep. 11, 2025. [Online]. Available: https://www.ncbi.nlm.nih.gov/pmc/articles/PMC9091082/