TUM Data Innovation Lab

Munich Data Science Institute (MDSI)

Technical University of Munich

&

TUM AI for Pathology, Schueffler Lab

Final report of project:

PathoAI Cockpit - Pathology Data Science for Molecular and Digital Pathology AI

Authors	Max Adam, Mei-Ling Fang, Kaan Kalaycioglu, Santhosh Kumar Ravi Kumar, Dilvan Sabir
Mentor(s)	Prof. Dr. Peter Schueffler, PD Dr. Katja Steiger, Dipl. Biol. Nicole Pfarr
Project Lead	Dr. Ricardo Acevedo Cabra (MDSI)
Supervisor	Prof. Dr. Massimo Fornasier (MDSI)

Jul 2023

Summary

Digital pathology has revolutionized diagnostic medicine, enabling efficient and accurate analysis of tissue samples through digitized slides. However, as digital pathology becomes more prevalent, maintaining quality control during the scanning process is crucial to ensure pathologists receive slides suitable for immediate analysis. In this project, we present the development of a comprehensive dashboard for digital pathology slides, integrating multiple artifact detection classifiers to address these challenges.

The frontend of the dashboard was built using the Angular framework, coupled with a backend divided into two components. The first component triggers upon slide scanning, calling the classifiers and collecting their outputs in JSON format. The second component serves as a REST API, delivering data to the frontend and facilitating a user-friendly interface for pathologists to access and interact with digitized slides effortlessly.

Python-based classifiers were developed to enhance quality control in digital pathology analysis, including:

1. **Air Bubble Detection:** Identifying slides with air bubbles that may hinder accurate analysis.
2. **Blur Detection:** Quantifying and detecting blur in slides to ensure clarity and reliability.
3. **Dust and Dirt Detection:** Identifying artifacts such as dust and dirt that could impact analysis.
4. **Missing Coverslip Detection:** Detecting slides without coverslips to ensure clarity and accurate analysis.
5. **Incomplete Tissue Detection:** Identifying areas where tissue samples are missing from the scanned image.
6. **Stripe Detection:** Detecting artifacts caused by scanning irregularities, ensuring slide accuracy.
7. **Tissue Fold Detection:** Identifying slides with tissue folds that could impact analysis.

The dashboard displays the results of each classifier in an easily interpretable manner, enabling a user to identify slides that might require rescanning or further examination. By integrating these classifiers into the dashboard, our project enhances the quality and efficiency of digital pathology analysis, addressing the growing need for reliable quality control measures in this evolving field.

Contents

1 Introduction

1.1 Digital Pathology

Digital pathology is a field of medicine that involves the digitization and analysis of pathology slides. It enhances the traditional method of manually examining glass slides under a microscope with the use of digitized Whole Slide Images (WSIs) that can be analyzed and interpreted using high-resolution scanners and computer algorithms. This enables remote access, collaboration, and automated diagnostics for improved accuracy and efficiency in diagnosing diseases.

1.2 Motivation

WSIs are produced in the digital pathology workflow (see Figure 1). A WSI is a high-resolution digital representation of an entire glass slide that is used in digital pathology for viewing, analyzing, and sharing tissue specimens electronically (see Figure 2b). With the adoption of advanced imaging tools in clinical and research practice, WSIs are used more and more to explore complex and sub-visual features in tissue-based biomarker analysis, which are related to different types of cancer. The need for increasing translational outcomes has sparked the innovation of a superior image analysis approach. Moreover, the enormous amount of data generated has made the management of diagnostic workflow increasingly hard. Therefore, there is an increasing need to develop tools to streamline the workflow of pathologists.

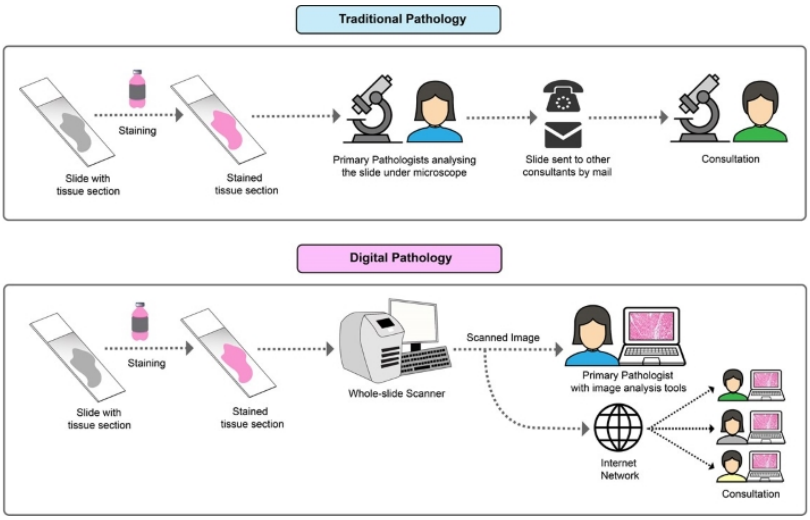


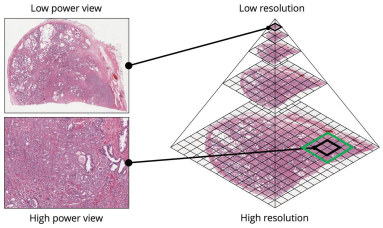
Figure 1: Comparison between the workflow of traditional and digital pathology [1].

Quality control is one of the most crucial bottlenecks in digital pathology. The lab[1] of our project mentor, Prof. Dr. Schueffler, is affiliated with the Institute of General and Surgical Pathology of the Technical University of Munich[2], where the digitization of WSI archives is taking place. In the current setup, the preparation and digitization of histological

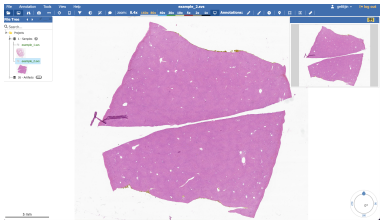
[1]Schueffler Lab
[2]Institute of General and Surgical Pathology of the Technical University of Munich

slides often introduce variations and artifacts, e.g. specks of dust (Figure 14a) or out-of-focus regions (Figure 5), but the detection of such artifacts are also manually screened using tools such as Figure 2c, which are time-consuming and error-prone. Artifacts are alterations of tissue or artificial structures introduced by extraneous factors that might be present in some parts or even the whole WSI [2], which might hamper the diagnostic procedure. This project focuses on the automation of detecting common artifacts of scanned images. In addition, we provide a solution for WSI quality control in the form of a cockpit dashboard, which integrates state-of-the-art implementations found in the literature. Furthermore, we include a summary of evaluation techniques along with a discussion of possible limitations and future refinement directions.

(a) WSI scanners



(b) Resolution pyramid



(c) Whole slide viewer

Figure 2: There are various vendors offering WSI scanners (2a and its corresponding format. Usually, the file consists of an image pyramid, with top-level showing a thumbnail image that has the lowest resolution and the bottom level of the pyramid (see 2b) having the highest resolution. Artifacts can occur at every resolution level. [3]. Pathologists can inspect WSIs using imaging tools such as the one in 2c.

2 Methods of Artifact Detection

In this section, we explain our approach to detecting the most common artifacts.

2.1 Air Bubbles

Air bubbles are confined air pockets that become trapped between the microscope slide and the cover slip. This is often a consequence of insufficient sealing of the sample.

Air bubbles pose a significant issue primarily due to their ability to cause light refraction, resulting in distortions within the scanned image. Consequently, the distorted image fails to faithfully represent the true characteristics of the tissue sample. This distortion then could pose a challenge in accurately interpreting tissue structures and features, in the worst case potentially leading to erroneous diagnoses or inaccurate analysis.

2.1.1 Related Work

During the research phase, we conducted a thorough literature review to identify suitable methodologies. We found one study that is of particular interest where the authors

(a) no bubbles (b) air bubble (c) air bubble at tissue

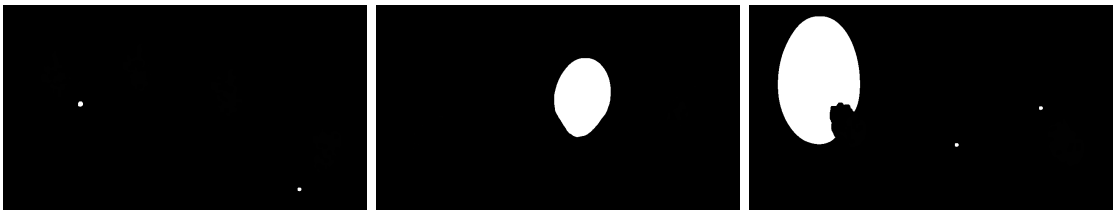
Figure 3: Example of no bubbles 3a, bubbles 3b, and bubbles intersecting with tissue 3c.

proposed and subsequently implemented an approach for air bubble detection that entails the iterative processing of the slide file to detect the contours of lines formed by air bubbles [4]. However, upon experimentation, we observed that this method lacked robustness, and the authors themselves acknowledged its developmental nature. In light of this, we sought guidance from our mentor, Prof. Dr. Peter Schüffler, who suggested an alternative strategy. This approach focuses on the detection of black lines, as they correspond to the boundaries of the bubbles. To explore this concept further, we consulted a publication by Schüffler et al. [5], which aimed to extract pen annotations of various colors (including blue, red, green, and black) from slide thumbnails.

2.1.2 Implemented Method

Concerning the presence and detection of air bubbles, our study began with a positive sample size of 4, which we augmented by implementing a black marker detector using OpenCV and Python. This detection method was primarily derived from the previous work conducted by Schueffler et al [5].

By employing the aforementioned approach, we successfully generated masks of the air bubbles by detecting the black lines encircling them and subsequently filling in the corresponding areas. Figure 4 demonstrates the generated masks for the samples from Figure 3:



(a) no bubbles (b) air bubble (c) air bubble at tissue

Figure 4: Masks of air bubbles of the images from Figure 3.

Furthermore, we developed a specialized tool utilizing the OpenSlide API to extract batches of thumbnail images from WSIs in the format of .svs files.

2.1.3 Results

We tested our classifier on a dataset of thumbnail images extracted from 3060 WSI from the TUM Institute of Pathology. Through meticulous manual data cleaning, we successfully identified 174 slides that displayed the presence of air bubbles. In order to avoid

lots of false positives, we set the masking sensitivity to pick up on objects of at least 300 pixels large, a number which was arrived at using grid-search.

We then constructed a Convolutional Neural Network (CNN) binary image classifier based on the TensorFlow framework. We employed a dataset consisting of 378 samples distributed across two classes: no bubbles (negative class) and bubbles (positive class). For training purposes, we utilized 320 samples, reserving 58 samples for testing.

In Table 1 and Table 2, we present the confusion matrix and the accuracy metrics respectively for the CNN air bubble classifier:

		Manual Detection		
		Bubbles	No bubbles	Total
Prediction	Bubbles	171	95	266
	No bubbles	49	2791	2840
	Total	220	2886	3060

Table 1: Confusion Matrix for the CNN Air Bubble Classifier on a dataset of 3060 WSIs.

Accuracy	**Precision**	**Recall**	**Specificity**	**F1-Score**
0.9494	0.6436	0.7778	0.9678	0.7043

Table 2: Accuracy metrics for the CNN Air Bubble Classifier on a dataset of 3060 WSIs.

2.1.4 Discussion

Limitations of this work include the constrained sample size employed for training the classifier. Consequently, the current classifier detects air bubbles in a general sense, rather than specifically focusing on the issue of bubbles intersecting or overlapping with tissue. Nonetheless, it is noteworthy that the latter scenario is a subset of the former, and thus the current classifier adequately fulfills its primary objective.

For future improvement, one potential avenue of approach involves integrating a segmentation model, such as Meta's Segment Anything [6] into the detection process. Proper implementation of this approach holds promise in achieving a robust method for accurately identifying instances where air bubbles overlap with tissue. This enhancement would offer a more precise and targeted detection capability, specifically addressing concerns related to bubble-tissue intersections.

2.2 Blurriness

2.2.1 Context

Blur is the most common artifact introduced during scanning that diminishes the overall sharpness of a WSI, see example slides in Figure 5. It is produced by uneven tissue thickness or improper focal calibration by the scanning devices [2]. Once blur areas are detected on the WSIs, they're often discarded or re-scanned if possible.

2.2.2 Related Work

Existing methods of objectively quantifying the presence of blurry patches can be divided into Full-/Partial-Reference and No-Reference methods [7]. Full-/Partial-Reference methods require a non-blurry reference image for comparison. However, reference images are not always available. In No-Reference methods, one assumes that the distribution of the blur metric is different in sharp and blurry patches.

Figure 5: Example slides from the FocusPath database, showing different levels of blurriness.

Gao *et al.* [8] leveraged the textural information obtained from the images and classify sharp and blurry regions by determining pixel-level information and bin distributions. Both local and global features are compared using several classifiers. Among them, local features contributes to higher accuracy. Campanella *et al.* [9] proposed to train a blur detector from scratch, using sharpness-based features along with a random forest model and residual network. The results include spatial heatmaps that enable further quantification and localization of blurred areas on a slide. Hosseini *et al.* [10] proposed a focus quality assessment metric by using a sum of even-derivative filter bases to synthesize a human visual system-like kernel, which is modeled as the inverse of the len's point spread function. Further, a benchmark database FocusPath [11] with 8640 pathological images (consisting of 9 slides from different organ tissues) was made available[3], and this dataset is used in this project for training and validation of a blur detection algorithm.

2.2.3 Implemented Method & Evaluation

Inspired by the work of Pech-Pacheco *et al.* [12], we implemented a blur detection method using a *variation of the Laplacian*. The idea is to highlight regions of an image containing rapid intensity changes by measuring the 2nd derivative of an image using the Laplacian operator. The assumption is that if an image has high variance, there is a wide range of responses, representative of an in-focus and out-of-focus image. On the other hand, low variance indicates there's a tiny spread of responses. Given that the more an image is blurred, the

$$\begin{bmatrix} 0 & 1 & 0 \\ 1 & -4 & 1 \\ 0 & 1 & 0 \end{bmatrix}$$

Figure 6: The Laplacian kernel.

[3]FocusPath-UofT https://sites.google.com/view/focuspathuoft/home

fewer edges there are. So, we start by convolving a grayscale image with the Laplacian kernel, see Figure 6. Then we take the variance, i.e. squared standard deviation, of the outcome. If the variance falls below a pre-defined threshold, then the image is considered blurry, and vice versa.

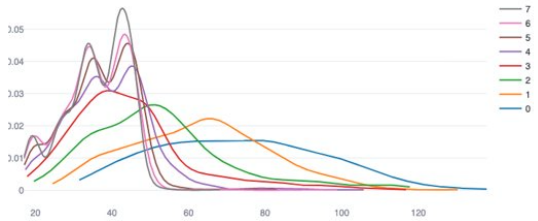
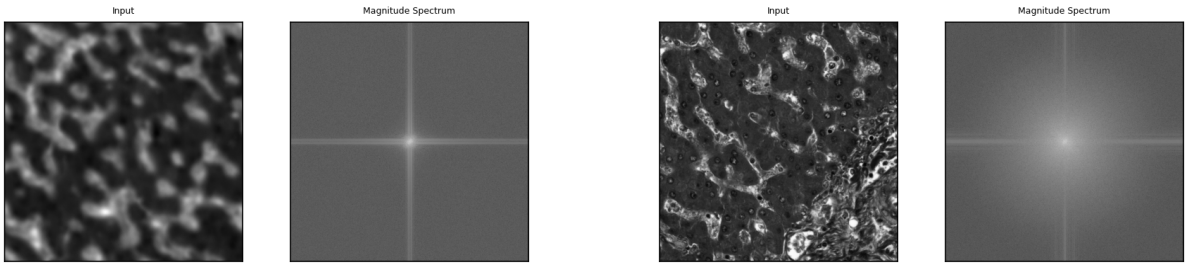


Figure 7: Distribution of image responses after applying Laplacian filter.

Next, we look for a clear threshold using the FocusPath [11] database. We start by classifying levels of blurriness into 7 categories, with high scores for blurry regions and low scores for sharp regions. Then, we plot the output of image responses after applying the Laplacian filter. From the distribution plot (Figure 7), we can observe a similar distribution among very blurry classes (score ≥ 4) and less blurry classes (score < 4). The color encodes the subjective score, which is defined in the FocusPath dataset. Here, we classify images with absolute subjective scores above 7 into 7, in order to balance the class distribution. However, due to the long tail distribution, we are unable to make a clear distinction between blurry and sharp images at this stage.
Then, we turn our attention to the 2nd technique - *Fast Fourier Transform*(FFT) [13,14]. It is an algorithm used to analyze signals and identify patterns in them. In blur detection, FFT is applied to an image to convert it into its frequency domain representation. By analyzing the high-frequency content in the spectrum obtained through FFT, we can determine whether the image is blurred or not based on the amount of high-frequency energy present. The magnitude spectrum image obtained from the FFT in blur detection represents the strength or magnitude of different frequency components present in an image. By converting the input image to grayscale and applying the FFT, we can analyze the distribution of frequencies in the image. This information helps to determine the level of blurriness in the image, as sharp images tend to have higher energy in the high-frequency components (see Figure 8b), while blurred images exhibit a more uniform distribution across frequencies (see Figure 8a).



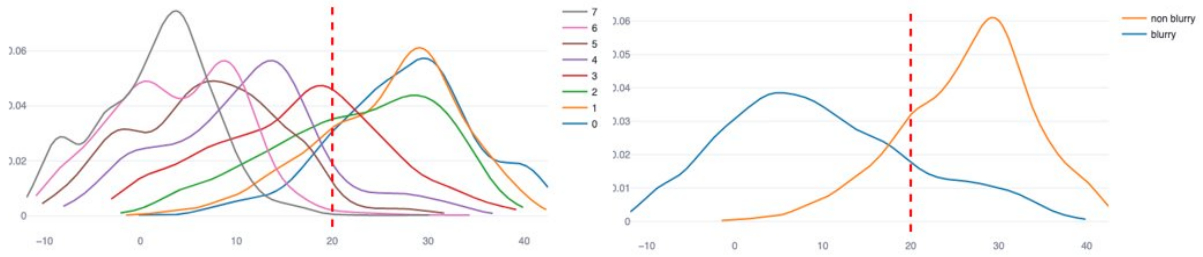
(a) Blurry image with subjective score of 8

(b) Sharp image with subjective score of 0

Figure 8: Comparison of blurry (a) and sharp (b) images after applying FFT. Users can find more information about subjective scores in [11]. The sharpest images get a score close to 0. The higher the score, the blurrier the image is.

To further inspect the effectiveness of our method, we applied the method on every patch

in the FocusPath database and plotted the distribution. Here, we empirically classify image patches with an absolute subjective score of 1 and 0 as sharp, and the rest as blurry. We set the threshold to 10. We subsequently obtained an acceptable separation between 2 classes, see Figure 9b. However, refinements and tuning the threshold are required for further improvement. At this stage, we present the threshold as well as the output values for users to decide.



(a) FFT (b) FFT with threshold

Figure 9: (a) Distribution of image responses after applying FFT and its corresponding blurry binary map (b).

The final result is demonstrated using an example WSI in Figure 10. In our implementation, the blur detector takes a WSI as input, then it generated tiles, or patches, in each layer and stores them into a *DeepZoom* object. It then takes the deepest level with the highest resolution and runs the above algorithms for each tile, and generated a heatmap visualization.

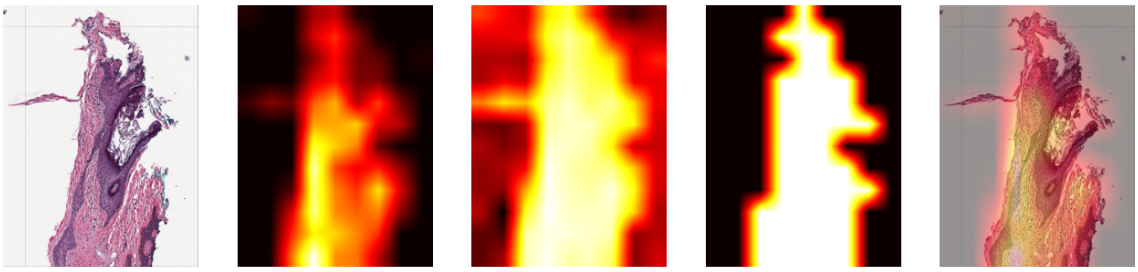


Figure 10: Final result of the blur detector. The leftmost image is the original image being detected. Then it follows the heatmap from the Laplacian filter, and the heatmap from FFT, as well as its binary blurry mask derived from FFT. The last image is the original image with an overlay of FFT heatmap. The lighter the region, the sharper it is.

2.2.4 Results

We continued to present deep learning (DL) as well as traditional machine learning (ML) methods, which were explored separately yet have not generated significant results due to time constraints, see the DL and ML path illustrated in Figure 11. These could be further improved for future explorations.

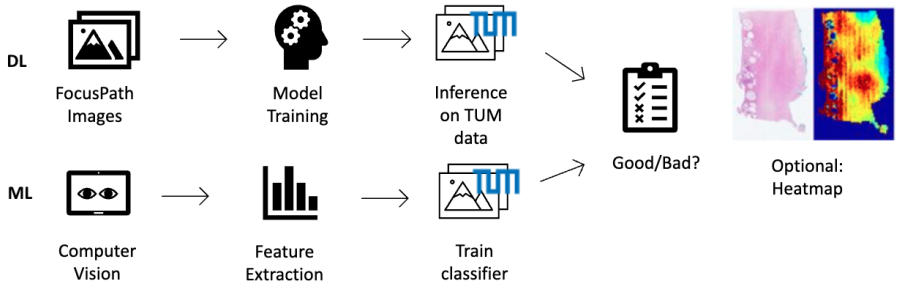


Figure 11: Proposed routes during the exploration phase.

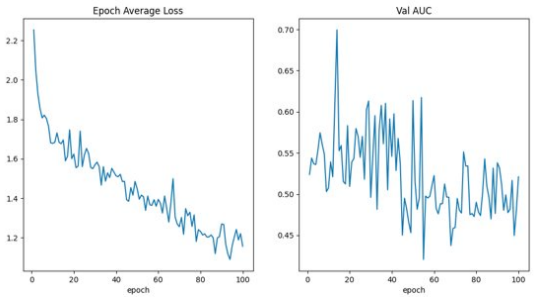


Figure 12: Training plot of DenseNet121.

In the DL path, a DenseNet121 multi-class classifier was trained using a subset of FocusPath, with a total of 864 images for the experiment. 4-fold split for cross-validation was performed, with a 90:10 split ratio for the training and testing set. The labels were taken from the subjective scores defined in the FocusPath. To balance the class distribution, we first took the absolute value of the subjective scores, and assigned scores above 9 into class 8, so that we obtained a total of 9 classes as our labels. Though the training average loss shows a clear declining trend Figure 12, the resulting classifier is barely outperforming a random one, see the performance in Table 3. We conjured that more preprocessing of the image patches, proper handling of the class labels, and lack of hyperparameter tuning might improve the predicting quality.

	AUC	**Weighted Precision**	**Weighted Recall**	**F1-Score**
Fold 0	0.6379	0.0625	0.0833	0.0833
Fold 1	0.6994	0.1738	0.1481	0.1191
Fold 2	0.6004	0.0258	0.0926	0.0926
Fold 3	0.6128	0.0981	0.1250	0.1250
Average	0.6376	0.0901	0.1123	0.1050

Table 3: Results of DenseNet121.

In the machine learning (ML) branch, a feature extraction method was explored. Inspired by [9], we would like to extract several textural, morphological, and histogram-based features from FocusPath datasets for training a random forest classifier to predict the blurriness score per patch. Due to computational constraints, the library[4] never completed the extraction of grayscale images from the full FocusPath datasets. However, we believe that the features from the image contain useful signals and would be helpful in training a supervised classifier.

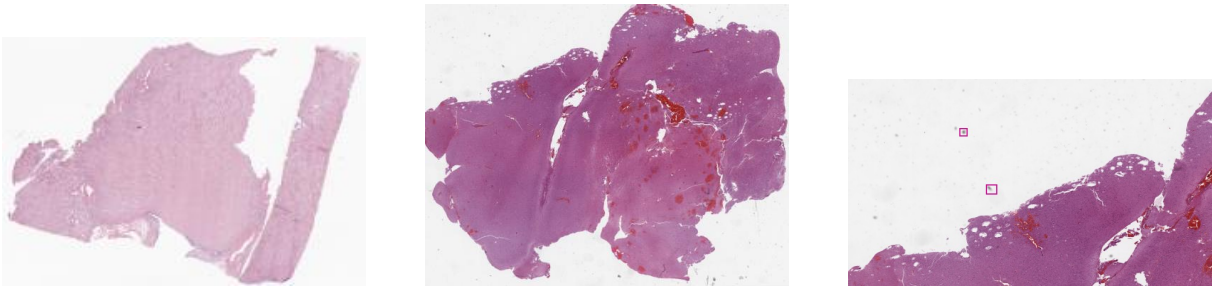
[4]pyfeats - Open source software for image feature extraction.

2.2.5 Discussion & Conclusion

We started development based on FocusPath [11] because it is hard to locate blurry slides among the available TUM datasets. The training and evaluation are based solely on FocusPath. Therefore, the transferability is yet to be examined. Furthermore, although the distribution plot was used to determine a sensible threshold, a more robust way to quantify the blurriness and determine a threshold should be considered in the future.

2.3 Dust & Dirt

Dust detection plays a crucial role in ensuring accurate analysis of digital pathological slides. Various techniques, including image processing algorithms and machine learning approaches, are employed to differentiate dust particles from tissue regions, improving the reliability of slide analysis.



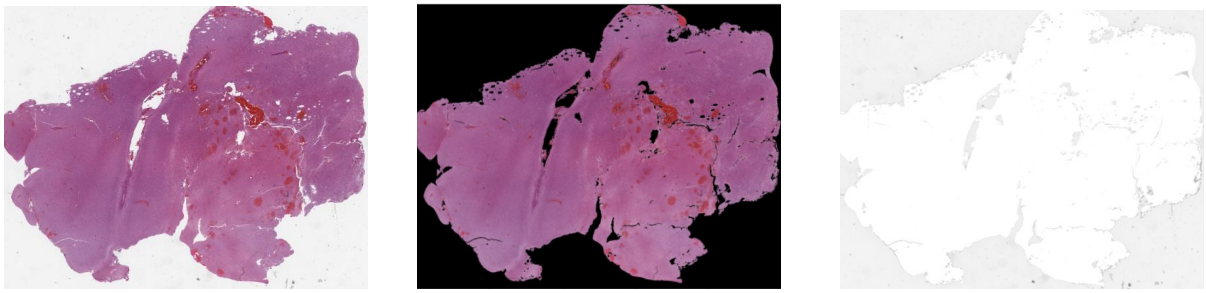
(a) Clean Slide (b) Significant dust present (c) marked dust

2.3.1 Overview

Many techniques could be deployed for detecting dust. Thresholding and morphological operations are commonly used to distinguish dust particles from the tissue regions. Texture analysis methods, such as local binary patterns or wavelet transforms, can further enhance detection accuracy by capturing the unique textural characteristics of dust particles. Machine learning approaches, including supervised and unsupervised methods, have also been utilized to train classifiers that can differentiate dust from other structures in the slide images. Integration of these techniques enables efficient and automated detection of dust, improving the quality and reliability of digital pathological slide analysis.

2.3.2 Implemented Method & Evaluation

Our proposed dust detection involves three key steps: image normalization, tissue masking, and thresholding. Initially, the image pixel values are normalized to a 0-255 range, ensuring consistent intensity values across the image as in Figure 14a. Subsequently, tissue masking is applied to isolate the tissue region as in Figure 14b, which is crucial for accurate dust detection. The resulting tissue mask is then converted to white Figure 14c, effectively highlighting the background while suppressing the tissue. Finally, a threshold value of 40 is applied to the entire slide, creating a binary image where pixel values below 40 represent potential dust points. This method assumes that dust is equally distributed



(a) Slide with dirt (b) Tissue alone masked (c) Tissue threshold to white

Figure 14: Step-by-step dust detection process. The original slide 14a contains both tissue and dust artifacts. In 14b, the tissue region has been masked. The tissue region 14c is blended while maintaining the dust artifacts on the outside region.

on the WSI and can be easily detected in the background.

2.3.3 Discussion

Our technique can be used for the classification of dust versus non-dust regions in digital pathological slides. However, it is important to note that the accuracy of the classification may depend on the specific characteristics of the dust particles and the complexity of the slide images. Fine-tuning or additional steps may be necessary to achieve higher classification accuracy, such as applying texture analysis, machine learning algorithms, or expert review for validation.

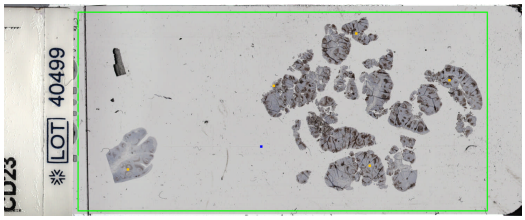
2.3.4 Conclusion

In conclusion, extending the current dust detection technique through region growth methods or CNN-based approaches represents potential avenues for future research. By iteratively expanding the regions based on certain criteria, such as intensity similarity or texture consistency, it may be possible to capture more subtle or fragmented dust artifacts that were initially missed. Furthermore, machine learning techniques offer an opportunity to enhance dust detection by training models on labeled dust areas. By generating a dataset consisting of annotated dust regions from pathological slides, supervised learning algorithms, such as convolutional neural networks (CNNs), can be trained to recognize dust patterns and discriminate them from other structures. These extensions could enhance the accuracy and completeness of dust detection in digital pathological slides. Addressing the associated challenges and limitations will be crucial for developing more advanced and reliable dust detection methods, ultimately contributing to improved slide analysis and diagnosis in the field of digital pathology.

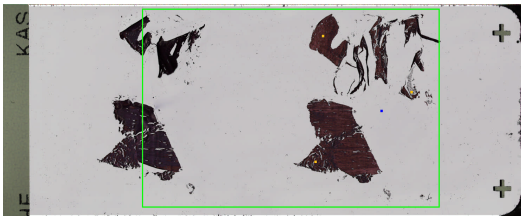
2.4 Missing Coverslip

Occasionally, some samples are scanned without their coverslips, posing a problem as the tissue appears significantly darker than its actual state, making the pathologist's task more challenging. Figure 15 illustrates the difference at the macro level of a typical slide

with a coverslip would resemble Figure 15a and an example of a slide without a coverslip in Figure 15b.



(a) With cover slide.



(b) Without cover slide.

Figure 15: Example macro images with coverslide present 15a and coverslide missing 15b.

An immediate distinction in Figure 15b as compared to Figure 15a is the absence of black lines or edges on the left and right sides of the slide. This characteristic is present only in the slides with a coverslip, as it is the boundary where the cover glass ends.

2.4.1 Related Work

To the best of our knowledge, there is no related work in the literature. With respect to other artifacts, it would appear that the occurrence of this issue is comparatively infrequent. We therefore subsequently chose to pursue our own approach focusing on classifying or clustering digital pathology images based on their color and brightness characteristics as well as the detection of the presence of black lines along the edges inside the slide, which might serve as reliable indicators of the presence of coverslip.

2.4.2 Implemented Method & Evaluation

(a) K-means of slide with coverslip.

(b) K-means of slide with no coverslip.

Figure 16: Result of K-means clustering.

Our study initially focused on identifying slides without coverslips, encompassing a positive sample size of 10. We explored various approaches, one of which involved analyzing the difference in brightness. Specifically, we attempted to cluster the slides based on dominant colors in the images using techniques such as K-means clustering as indicated in Figure 16. However, this method proved unreliable due to the substantial variance within images outweighing the variance between groups.
Subsequently, we adopted a second and final approach that focused on detecting the black lines on the right and left sides of the slides. This was accomplished through the utilization of OpenCV.

We performed hyperparameter tuning and adjusted the parameters of a custom-written black line detector in OpenCV using an initial sample size of 177 slide images. Within this sample, 10 belonged to the positive class (slides without coverslips), while the remaining samples represented the negative class (slides with coverslips). Through rigorous optimization, the detector achieved flawless classification accuracy, without any misclassifications.

2.4.3 Results

We applied this optimized detector to an expanded dataset consisting of 3053 slide images. Impressively, it accurately identified 2989 slide images as having coverslips, resulting in zero false negatives. Within the positive class, however, the detector detected 64 images, of which 61 were false positives, and 3 were true positives.

		Manual Detection		
		Positive	Negative	Total
Prediction	Positive	3	61	64
	Negative	0	2989	2989
	Total	3	3050	3053

Table 4: Confusion Matrix for the coverslip detector on a dataset of 3060 WSIs.

Accuracy	**Precision**	**Recall**	**Specificity**	**F1-Score**
0.9812	0.0462	1.0	0.9801	0.0898

Table 5: Accuracy metrics for the coverslip detector on a dataset of 3053 WSIs.

2.4.4 Discussion

A potential direction for future improvement in missing coverslip detection involves revisiting the analysis of brightness and colors within macro images. By considering the commonalities and differences in colors between the positive class (coverslip absent) and negative class (coverslip present), it may be possible to effectively cluster new images based on these criteria. A clustering approach could help address the challenge of limited samples for the positive class, assuming one can accurately capture the features of the macro images that differentiate the positive class from the negative class.
Another avenue to explore is the detection of vertical black lines on the left and right sides of the macro image. However, it is important to consider that some slides may contain a black region on the right side as a consequence of the scanning process or because specific information such as the slide's make or the manufacturer's name, which is typically presented as white text is on a black background on the right side of the macro image. Any approach incorporating black line detection should account for these potential confounding factors.
Furthermore, a promising prospect may also involve combining these two approaches. By integrating the analysis of brightness and colors with the detection of vertical black lines,

it might be plausible to develop a highly reliable and potent method for missing coverslip detection in digital pathology.

2.5 Incomplete Tissue

During the scan process, the scanner defines a bounding-box around the detected tissue. Occasionally, the scanner fails to correctly locate the tissue which results in an incomplete scan, meaning parts of the tissue on the WSI are not captured. In that case, a re-scan has to be triggered. In general, we distinguish two cases of incomplete tissue: touching tissue and missing tissue.

2.5.1 Implemented Method & Evaluation

Touching Tissue Detection: If contiguous tissue has only been captured in parts by the scanner, there is tissue directly at the edge of the scan (Figure 17). In this case, the automatic detection is straightforward. We use a method for tissue detection introduced by Wang [15] to pixelwise segment the scan into two classes 'tissue' and 'no tissue'. Then, we calculate the percentage of tissue that is present in the four-pixel-wide border region of the scan. The value of four was chosen because we only want to capture tissue that is immediately at the edge of the scan to minimize the risk of falsely classifying tissue as a cut-off. The resulting score indicates the relative amount of border that has touching tissue, with zero percent meaning none of the border region has tissue while 100 percent means the entire border region has tissue.

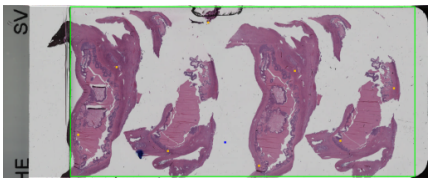


Figure 17: Example of touching tissue. Green: scanned region.

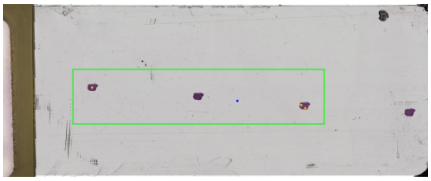


Figure 18: Example of missed tissue. Green: scanned region.

Missing tissue detection: In the instance that tissue is non-contiguous, tissue that is left out by the scanner can not be detected on the scan itself (Figure 18). For that case, we have to analyze the so-called macro image for tissue outside the bounding box. The macro image is a picture of the scan slide taken by the scanner during the scan process. However, classical tissue detectors are trained on the scan itself, not on a picture of the scan slide. Therefore, our tests have shown that those models are not suitable for reliably detecting tissue using the macro image, as they tend to confuse macro-image-specific artifacts with tissue.

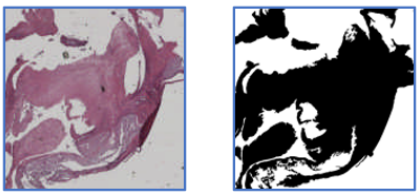


Figure 19: An auto-generated patch of the macro image with its corresponding tissue mask.

Model training: Since we have a large dataset of tissue scans that each include a macro image and the bounding box, we can automatically generate a labeled dataset to train our own macro image tissue segmentation model. Therefore, we first segment the tissue on the

WSI using the method from [15]. Then, we use the location and size of the bounding box extracted from the scan metadata to project the tissue mask onto the macro image. This results in a macro image segmented into three classes; 'tissue', 'no tissue', and 'unknown'. The latter represents the area outside the bounding box, where we do not know if there is tissue or not since this area was not captured by the scanner. After discarding the unknown part of the segmented macro image, we sample 20 100x100 pixel patches from each macro image, consisting of a macro image section together with a binary tissue mask (Figure 19). In total, we generated 8000 patches from 400 macro images. Since the scanner does not scan the label area and the dark edges of the slide that can both be seen in Figure 17 and Figure 18, there are no labeled samples of those regions in our dataset. Therefore, we added 200 manually sampled patches from those edge regions. The 8200 labeled patches are then used to train a UNet model for the task of macro image tissue segmentation. The model architecture was inspired by [16] and consists of a conventional encoder for downsampling, a bottleneck consisting of a convolutional block, and a decoder for upsampling. We use a random 80/10/10 train/test/val split and trained our model using binary-cross-entropy-loss and a learning rate of e^{-3}. Since the majority of the patches contain none or only little tissue, our training dataset is highly imbalanced. Therefore, we use the F1 score as an evaluation metric which can handle imbalanced datasets. We trained our model for 20 epochs and kept the model weights that achieved the best F1 score on the validation set to run evaluation on our test set. The training took 15 minutes on a Tesla T4 GPU. The trained model was able to achieve an F1 score on the test set of 0.9904.

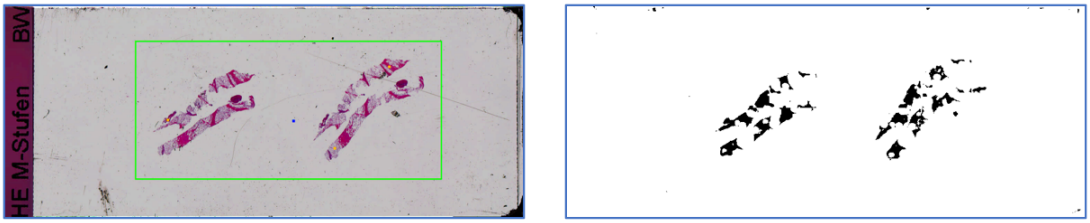


Figure 20: A sample macro image segmented by our macro image tissue segmentation model.

Model inference: Since our model is only able to segment 100x100 pixel macro image patches, we have to divide new macro images into patches of this size and process them individually. Figure 20 shows a sample macro image segmented using our model. Although small spots at the edge of the image are incorrectly labeled as tissue, the model is able to correctly identify the tissue present. With the location and size of the bounding box, we can then calculate the percentage of tissue that is located outside the bounding box and thus not included in the scan.

2.5.2 Conclusion

Detecting touching tissue is a simple and straightforward task that can be solved with simple numerical algorithms. However, to successfully detect missing tissue, we had to

train our own macro image tissue segmentation model. We chose the UNet architecture because our literature review showed UNet to be very suitable for image segmentation tasks [17]. Although we do not have a dataset of fully labeled macro-images that could be used for robust evaluation, visual inspection of fully segmented samples shows that our approach already works pretty well. More training samples and extensive hyperparameter tuning could lead to even better performance. In addition, our model can currently only be as good as the WSI tissue segmenter we use to generate training samples. In addition, the way we patch the macro images both for the creation of the training dataset and during inference can still be optimized.

2.6 Stripes

2.6.1 Introduction

One common artifact for WSI is the presence of stripes, which can arise from various sources such as scanner imperfections, tissue folds, dirt/dust, or staining irregularities. Stripe artifacts exhibit distinct illumination variations due to differences in the white balancing procedure in the scanner. (Figure 21) The existence of stripes poses a significant challenge to downstream image analysis algorithms, as they can introduce bias and distort the extracted features, potentially leading to erroneous interpretations. Therefore, the development of reliable and efficient stripe detection techniques is crucial to ensure the integrity and accuracy of digital pathology workflows. We are proposing an algorithm based on various image-processing techniques for detecting stripes in slides.

Figure 21: Example of stripes in a WSI.

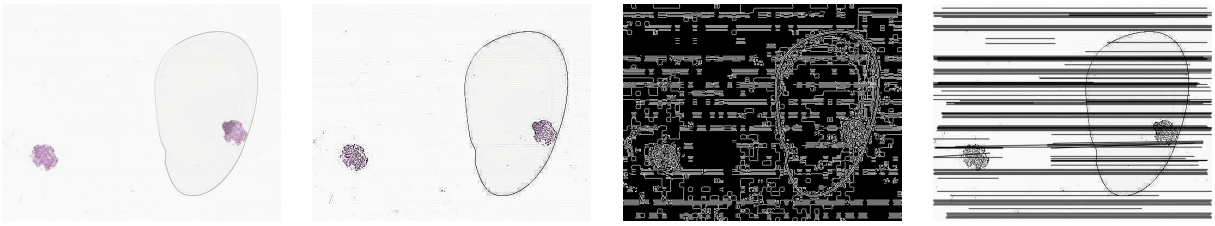
2.6.2 Related Work

It is possible to handle the problem of stripe detection using different approaches such as image-processing or deep learning. One approach by [18] makes use of the periodic nature of the stripe noise. Initially, the authors employ the direct and inverse fuzzy transform of the spectrum to identify periodic noise peaks. Subsequently, they introduce a notch filter based on the fuzzy transform to smoothen the spectral data and isolate the original image by removing the periodic noise elements.
Various other solutions (eg., [19]) make use of the feature extraction of CNNs. They propose a more general framework for slide segmentation and analysis. Their framework, in essence, is a series of individual techniques within the preprocessing-training-inference pipeline to enhance the efficiency and overall applicability of the analysis.
While above-mentioned approaches have shown promising results, they often require significant computational resources and may not generalize well to different types of stripes and image datasets. Therefore, there is a need for further research to develop robust,

efficient, and adaptable methods for stripe detection on whole slide images, considering the wide range of potential stripe sources and their varying characteristics.

2.6.3 Implemented Method & Evaluation

In our solution, we combine different image processing algorithms such as Canny edge detection, Gaussian blur, and Hough line transform to come up with a result. Our main idea is to combine the periodic nature of the stripes with edge detection techniques to obtain the number of significant horizontal lines that correspond to stripes in whole slide images. For this purpose, we use the thumbnail image of slides.
In our pipeline, we first apply an edge enhancement filter implemented in the Python Image Library (PIL) which applies an edge detection kernel to our image using convolution. Then we convert the image to grayscale format in order to prepare it for the edge detection algorithm. After that we apply the Canny Edge Detection algorithm which can be summarized in 5 steps: 1) Apply Gaussian filter to smooth the image in order to remove the noise, 2) Find the intensity gradients of the image, 3) Apply gradient magnitude thresholding or lower bound cut-off suppression to get rid of spurious response to edge detection, 4) Apply double threshold to determine potential edges, 5) Track edge by hysteresis: Finalize the detection of edges by suppressing all the other edges that are weak and not connected to strong edges.
This algorithm takes a grayscale image as input and returns its edge map using three parameters: two threshold values for the hysteresis procedure and the size of the Gaussian filter. The resulting edge map undergoes the Probabilistic Hough Transform, an extension of the Hough Transform for detecting line segments in an image. This step reduces computational complexity, ensures precise localization, and handles noise or occlusions well. The Hough transform algorithm has additional parameters (rho, theta, threshold, minLineLength, and maxLineGap) that require manual tuning for our use case. We will optimize minLineLength and maxLineGap, which will help classify slides based on the number of horizontal lines detected in the image. Figure 22 showcases the effect of different pipeline stages on a thumbnail image with stripes.



(a) Original image (b) Enhanced edges (c) Canny edges (d) Horizontal lines

Figure 22: Different stages of the stripe detection algorithm.

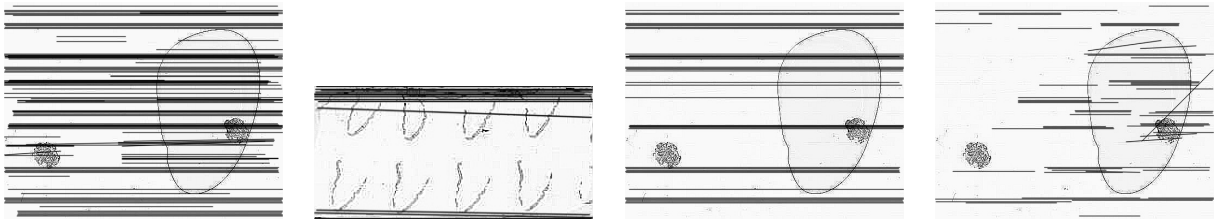
2.6.4 Results and Discussion

We selected 7 whole slide images from our dataset of slides to test how our algorithm performs. We manually tuned our parameters such that we get more consistent results. Values that we used for the parameters are as follows: Gaussian kernel size = 5, first

threshold for canny = 50, second threshold for canny = 150, rho = 1, theta = $\pi/180$, threshold = 100, minLineLength = 100, maxLineGap = 80. minLineLength is the minimum length in which lines below this length are rejected and maxLineGap is the maximum allowed gap between points on the same line to link them. We selected these two parameters so that it captures the stripes in our images more consistently. Figure 23 and Table 6 illustrates example results and the effect of the change of parameter values.

WSI ID	detected edges
2427	240
149651	101
171070	217
194606	52
362566	160
149657	33
non-stripe	12

Table 6: Stripe detection results.



(a) (b) (c) (d)

Figure 23: 23a Stripe slide with number of horizontal lines = 101
23b Normal slide with number of
23c Stripe slide with number of horizontal lines = 37 when minLineLength = 500s
23d Stripe slide with number of horizontal lines = 57 when maxLineGap = 10

2.6.5 Conclusion

We focused on addressing the challenge of detecting stripes in WSI, that can introduce bias and distort features in automated image analysis. To tackle this issue, we proposed an algorithm that employed image processing techniques, specifically combining edge detection and the probabilistic Hough transform to identify line segments associated with stripes. The effectiveness of our approach was evaluated on a dataset of eight whole slide images, and we manually adjusted the algorithm's parameters to ensure consistent and accurate results. Compared to alternative methods, our approach offers an advantageous combination of computational efficiency and adaptability, making it suitable for various staining conditions and slide characteristics. However, certain challenges persist, such as accounting for the diverse appearances of stripes, optimizing parameter selection, and establishing standardized benchmark datasets for rigorous evaluation.

2.7 Tissue Folds

2.7.1 Introduction

The presence of tissue folds, which occur due to mechanical problems during slide preparation, poses a significant challenge to accurate and reliable image analysis. Tissue folds can introduce distortions and artifacts that can compromise the interpretation of histopathological features, leading to erroneous diagnoses and unreliable quantitative measurements. See Figure 24 for an example of tissue fold. Consequently, the development of robust and efficient techniques for tissue fold detection on whole slide images has become a critical research area. In order to tackle this problem, we decided to take the method developed by [20] and adapt it to our use case.

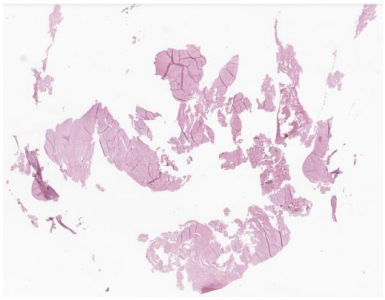


Figure 24: Example of tisse fold in WSI.

2.7.2 Related Work

In recent years, various methods have been proposed for tissue fold detection on WSI, including image processing, machine learning, and deep learning approaches. Early works focused on connectivity-based soft thresholding [21] and using HSV color space properties to detect changes in saturation [20]. Deep learning techniques have also gained attention, with [22] introducing a CNN architecture specifically designed for tissue fold detection. Their model combined CNN's ability to learn hierarchical features with an SVM classifier, achieving good accuracy and robustness. Despite these promising techniques, challenges remain in tissue fold detection due to variability in appearance influenced by size, shape, and texture. Additionally, the lack of publicly available annotated datasets hampers approach comparison and benchmarking. Overcoming these challenges will enhance accuracy and reliability in whole slide image analysis, benefiting digital pathology as a whole.

2.7.3 Implemented Method & Evaluation

The colorimetric characteristics of tissue folds can vary across slides with different staining conditions; however, they generally exhibit higher color saturation compared to non-fold regions. To effectively enhance the colorimetric distinction between tissue folds and other tissue components, the technique that we employed involves the shifting of RGB color values. The method also introduces an adaptive shifting factor that varies in magnitude, assigning a larger magnitude to pixels that are more likely to belong to tissue folds. To determine these pixels, we utilized the information present in the luminance and saturation

components of each image pixel. Equations 1 and 2 below show the modifications of luminance and saturation components that results in enhanced tissue fold regions in the overall image.

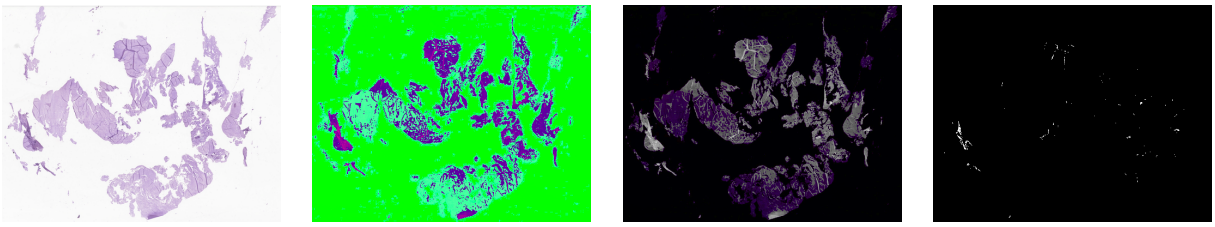
$$S' = 1 - 3\frac{min(R, G, B) + (S - V)}{R + G + B + 3\alpha(S - V)} \tag{1}$$

$$V' = V + \alpha(S - V) \tag{2}$$

Where S = saturation, V = luminance, R = red, G = green, and B = blue component of the corresponding pixel, and α is the shifting factor. The change in luminance becomes positive when $\alpha > 0$ and $S > V$. This change in luminance will be important when we are identifying the tissue folded regions in WSI. After applying this technique, we masked the image so that if the lumination value of the pixel is increased we set its value to 1 and 0 otherwise. This results in a binary mask where tissue folded regions can be seen as white. As a classification result, we decided to use the total number of white pixels. We also calculated the folded region area and the total tissue region in order to provide the users of the pathology application with more information since only the number of folded pixels may not be very meaningful.

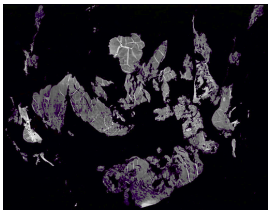
2.7.4 Results and Discussion

We selected 10 WSI from the dataset that we manually identified to contain tissue folds. Then we applied the pixel enhancement method to the thumbnail images that we extracted from the slide files. The paper that inspired our solution mentions that the alpha parameter should be selected between 1 and 2 (i.e. $1 < \alpha < 2$). They suggested that the optimal value for the alpha is 1.5 but in our experiments, we saw that even though $\alpha = 1.5$ works perfectly for most of the cases, increasing it resulted in better samples where the colors are less brighter. The steps of the algorithm and results for a single image can be seen in Figure 25 and Figure 26 respectively.



(a) original image (b) HSV color space (c) enhanced (d) binary

Figure 25: Steps of the tissue fold detection algorithm.



(a) enhanced

(b) binary

Figure 26: 25a with $\alpha = 1.8$ resulting in 3.24% of fold area and 1563 folded pixels.

See Table 7 for experimental results on a WSI which is known to contain tissue fold.

Slide No	Fold Area (cm2)	Tissue Area(cm2)	Ratio(%)
1	0.041	1.272	3.238
2	0.363	3.264	11.120
3	0.119	0.389	30.526
4	0.112	0.627	17.863
5	0.019	0.822	2.351
6	0.115	0.965	11.946
7	0.0832	0.116	71.742

Table 7: Some sample results of fold detection algorithm on WSI with folds.

2.7.5 Conclusion

We addressed the challenge of tissue fold detection on whole slide images by adapting a method proposed by Bautista et al [20]. Our implemented method focused on enhancing the colorimetric distinction between tissue folds and other tissue components by modifying the luminance and saturation components of each pixel. By shifting the RGB color values based on these modifications and masking the image, tissue fold regions were highlighted as white. We evaluated our method on a dataset of 10 whole slide images and observed that increasing the shifting factor alpha beyond the recommended value of 1.5 produced better results, especially for samples with less bright colors. Our method offers a simple yet effective solution for tissue fold detection, which is computationally efficient and easily adaptable to different staining conditions. However, challenges remain in addressing the variability in tissue fold appearance and the absence of standardized benchmark datasets for evaluation.

3 PathoAI Cockpit and Integration

The main purpose of our dashboard is to list WSIs that have been processed by the classifiers that we implemented. Here we present our architectural design as well as showcase our first version of the quality control dashboard. We further discuss the challenges, the limitations, and the potential improvements in the future.

3.1 Architectural Design

3.1.1 Backend

Our backend is composed of two main parts. One is the script that is invoked when a new slide has been scanned. This script runs all the classifiers and algorithms that were implemented for artifact detection and puts a resulting JSON file in an output directory. The second part is a REST API that serves as a single endpoint for the purpose of delivering artifact detection results to the frontend application. All scripts are written in Python 3.10 and use Flask as an API framework.

3.1.2 Frontend

We use the Angular framework v15 [23] for our frontend application. The Angular project follows a standard directory structure with the main components organized in the app/ directory and services in the app/services directory.
Each component is contained in its own folder with TypeScript, HTML, and optional CSS files. The application uses Angular Router for navigation, and the routing configuration is defined in the app-routing.module.ts file.
Global styling is managed through styles.css, and component-specific styles are defined in their respective CSS files.

3.2 Demonstration

Our PathoAI Cockpit runs locally on the TUM Slide Viewer server. A user can see the path of the WSI in the server, the image ID, outputs of the classifiers, thumbnails, and macro images. We also add a hyperlink to each slide that takes the user to the TUM Slide Viewer. We want to keep the layout basic and easy to interpret. A screenshot of the dashboard can be seen in Figure 27. There is room for further visual improvements as our focus in this project was strictly on functionality. And also, performance of the application should be evaluated in a production environment with many WSI.

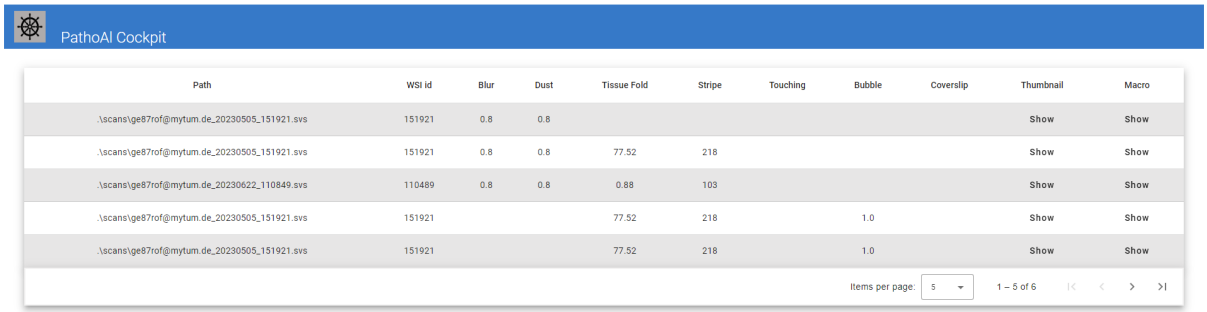


Figure 27: Screenshot from our quality assurance dashboard.

4 Discussion

Quality control (QC) evaluation has shown how artifact detection and data curation affect the performance of computational pathology systems [24,25]. However, artifact detection is still frequently overlooked in the preprocessing pipeline. Current approaches often rely on low magnification analysis to discard complete WSIs, including most of our detection algorithms. To effectively address this limitation, it might be necessary to extend QC approaches to higher magnification levels for artifact detection.
The massive size of the WSIs also poses a great challenge to the existing infrastructures. Moreover, the complexity of preprocessing required for WSI analysis, which varies based on tissue type, disease type, and specific tasks [2], makes selecting an appropriate preprocessing pipeline a daunting task. Our dashboard offers a tissue- and disease-agnostic solution, serving as the initial checkpoint and triage in the preprocessing pipeline for pathologists, effectively reducing manual workload. Nevertheless, further refinement of the detection algorithms is warranted, and cautious interpretation of results is essential.
Moreover, artifact detection can impact subsequent processing steps, such as color normalization and image augmentation. Whether to perform artifact detection before color normalization or not remains an open question. Additionally, detecting specific artifacts often depends on tissue characteristics, biopsy features, and laboratory procedures. In this project, we assumed the independence of artifact detection from such impacts in our workflow.
Through the integration of artifact detection and the development of a basic but comprehensive dashboard, our project aims to enhance the reliability and efficiency of digital pathology. By addressing these challenges, we contribute to advancing digital pathology's potential for accurate and automated analysis in diagnostic medicine.

5 Conclusion

In this project, we set out to enhance and streamline the digital pathology workflow by introducing a QC cockpit dashboard with artifact detection capabilities. This includes the implementation of various detection algorithms targeting prevalent artifact types within WSIs as well as evaluating them using the appropriate metrics and discussing their limitations.
The presented dashboard serves as a promising initial step toward optimizing digital pathologists' tasks. Moving forward, it is possible to further augment the functionality of the cockpit without compromising existing features and performance. This could entail incorporating valuable feedback from pathologists to introduce more useful features and further refine the detection speed. Additionally, expanding the range of detectable artifacts, addressing edge cases, and integrating scanner status and workload statistics are paramount to enhancing the dashboard's overall effectiveness and impact.

References

[1] C. A. Jahangir, "Promises of tissue based imaging in cancer research and diagnostics," *Precision Oncology Ireland*, 2020. [Online]. Available: https://www.precisiononcology.ie/newsevents/blogs/items/text,502846,en.html

[2] M. Ó. Cinnéide and P. Fagan, "Design patterns: the devils in the detail," in *Conference on Pattern Languages of Programs*, 2006.

[3] D. S. McClintock, J. T. Abel, and T. C. Cornish, "Whole slide imaging hardware, software, and infrastructure," *Whole Slide Imaging*, 2021.

[4] Y. Chen, J. Zee, A. Smith, C. Jayapandian, J. Hodgin, D. Howell, M. Palmer, D. Thomas, C. Cassol, A. B. Farris, K. Perkinson, A. Madabhushi, L. Barisoni, and A. Janowczyk, "Assessment of a computerized quantitative quality control tool for whole slide images of kidney biopsies," *J Pathol*, vol. 253, no. 3, pp. 268–278, Mar 2021.

[5] P. Schüffler, "Mskcc-computational-pathology/penannotationextractor: Pen annotation extraction from wsi," Feb 2021. [Online]. Available: https://github.com/MSKCC-Computational-Pathology/PenAnnotationExtractor

[6] A. Kirillov, E. Mintun, N. Ravi, H. Mao, C. Rolland, L. Gustafson, T. Xiao, S. Whitehead, A. C. Berg, W.-Y. Lo *et al.*, "Segment anything," *arXiv preprint arXiv:2304.02643*, 2023.

[7] H. Wu, J. H. Phan, A. K. Bhatia, C. A. Cundiff, B. M. Shehata, and M. D. Wang, "Detection of blur artifacts in histopathological whole-slide images of endomyocardial biopsies," *2015 37th Annual International Conference of the IEEE Engineering in Medicine and Biology Society (EMBC)*, pp. 727–730, 2015.

[8] D. Gao, D. R. Padfield, J. Rittscher, and R. McKay, "Automated training data generation for microscopy focus classification," *Medical image computing and computer-assisted intervention : MICCAI ... International Conference on Medical Image Computing and Computer-Assisted Intervention*, vol. 13 Pt 2, pp. 446–53, 2010.

[9] G. Campanella, A. R. Rajanna, L. Corsale, P. J. Schüffler, Y. Yagi, and T. J. Fuchs, "Towards machine learned quality control: A benchmark for sharpness quantification in digital pathology," *Computerized medical imaging and graphics : the official journal of the Computerized Medical Imaging Society*, vol. 65, pp. 142–151, 2017.

[10] M. S. Hosseini, J. A. Z. Brawley-Hayes, Y. Zhang, L. Chan, K. N. Plataniotis, and S. Damaskinos, "Focus quality assessment of high-throughput whole slide imaging in digital pathology," *IEEE Transactions on Medical Imaging*, vol. 39, pp. 62–74, 2018.

[11] M. S. Hosseini, Y. Zhang, and K. N. Plataniotis, "Focus quality metric based on visual sensitivity," *arXiv: Image and Video Processing*, 2018.

[12] J. L. Pech-Pacheco, G. Cristóbal, J. Chamorro-Martínez, and J. Fernández-Valdivia, "Diatom autofocusing in brightfield microscopy: a comparative study," *Proceedings 15th International Conference on Pattern Recognition. ICPR-2000*, vol. 3, pp. 314–317 vol.3, 2000.

[13] A. W. R. Fisher, S. Perkins and E. Wolfart., "Fourier transform," 2003. [Online]. Available: https://homepages.inf.ed.ac.uk/rbf/HIPR2/fourier.htm

[14] A. Rosebrock, "Opencv fast fourier transform (fft) for blur detection in images and video streams," 2020. [Online]. Available: https://pyimagesearch.com/2020/06/15/opencv-fast-fourier-transform-fft-for-blur-detection-in-images-and-video-streams/

[15] D. Wang, A. Khosla, R. Gargeya, H. Irshad, and A. H. Beck, "Deep learning for identifying metastatic breast cancer," *arXiv preprint arXiv:1606.05718*, 2016.

[16] Milesial, "Pytorch-unet," https://github.com/milesial/Pytorch-UNet/tree/master, 2021.

[17] H. Huang, L. Lin, R. Tong, H. Hu, Q. Zhang, Y. Iwamoto, X. Han, Y.-W. Chen, and J. Wu, "Unet 3+: A full-scale connected unet for medical image segmentation," in *ICASSP 2020-2020 IEEE international conference on acoustics, speech and signal processing (ICASSP)*. IEEE, 2020, pp. 1055–1059.

[18] N. Alibabaie and A. Latif, "Adaptive periodic noise reduction in digital images using fuzzy transform," *Journal of Mathematical Imaging and Vision*, pp. 1–25, 2021.

[19] M. Khened, A. Kori, H. Rajkumar, B. Srinivasan, and G. Krishnamurthi, "A generalized deep learning framework for whole-slide image segmentation and analysis," 2020.

[20] P. A. Bautista and Y. Yagi, "Improving the visualization and detection of tissue folds in whole slide images through color enhancement," *Journal of Pathology Informatics*, vol. 1, no. 1, p. 25, 2010. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S2153353922001171

[21] S. Kothari, J. Phan, and M. Wang, "Eliminating tissue-fold artifacts in histopathological whole-slide images for improved image-based prediction of cancer grade," *Journal of pathology informatics*, vol. 4, p. 22, 08 2013.

[22] M. Babaie and H. Tizhoosh, *Deep Features for Tissue-Fold Detection in Histopathology Images*. Springer International Publishing, 07 2019, pp. 125–132.

[23] [Online]. Available: https://angular.io/

[24] R. Wetteland, K. Engan, T. Eftestøl, V. Kvikstad, and E. A. M. Janssen, "A multiscale approach for whole-slide image segmentation of five tissue classes in urothelial carcinoma slides," *Technology in Cancer Research & Treatment*, vol. 19, 2020.

[25] A. Janowczyk, R. Zuo, H. Gilmore, M. D. Feldman, and A. Madabhushi, "Histoqc: An open-source quality control tool for digital pathology slides." *JCO clinical cancer informatics*, vol. 3, pp. 1–7, 2019.