WSI segmentation

2.1 Are portions of the slide background detected as tissue?

When using the fast segmentation model 'grandqc' instead of 'hest', the model has more difficulty distinguishing tissue from background, resulting in a larger portion of the background being misclassified as tissue. This illustrates a trade-off between speed and accuracy: the faster the model performs segmentation, the lower the quality of the classification. This is likely because the fast model was trained on a simpler dataset, designed for less complex tasks, and therefore is less precise. In contrast, 'hest' is slower but provides more accurate segmentation results. An example of this difference can be seen in Figure 1A,B. Nevertheless, the fast model 'grandqc' can outperform 'hest' in certain cases. For instance, in Figures 1C,D, 'hest' fails to correctly classify brighter tissue regions, whereas 'grandqc' succeeds in detecting them, although it also introduces a larger amount of background.

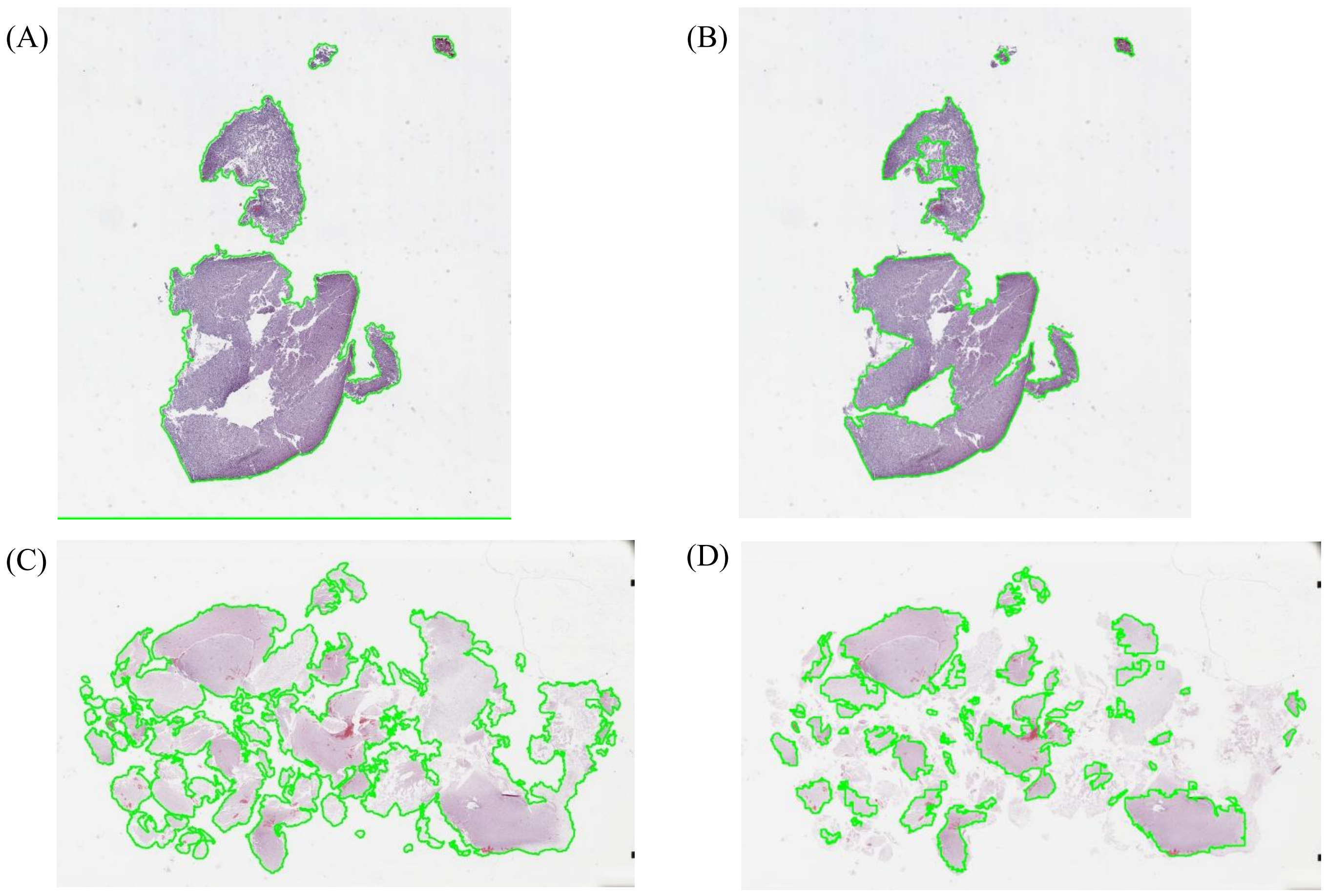


Figure 1. Example of differences between the models. (A,C) 'grandqc'; (B,D) 'hest'

In addition, even with the more complex 'hest' model, small portions of the slide background are sometimes misclassified as tissue. While 'hest' is generally more accurate than 'grandqc' and better at distinguishing tissue from background, it is not perfect. For instance, if there are voids within a tissue mass (such as a small circular patch of background embedded within a larger tissue region; Figure 2) the model may fail to recognize these as non-tissue.

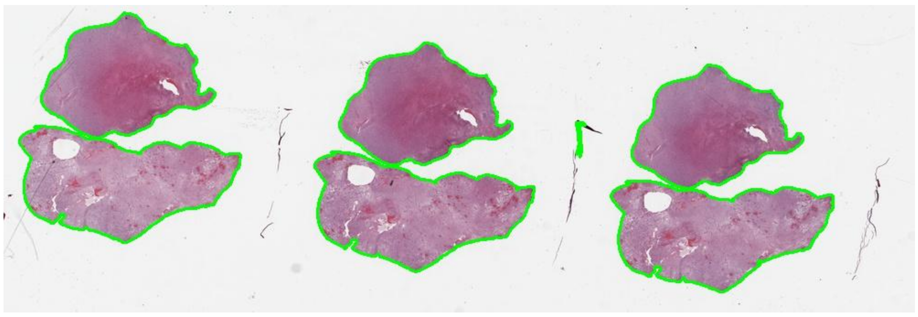


Figure 2. Example of slide background regions incorrectly detected as tissue

2.2. Are portions of tissue not detected?

Yes, certain tissue regions were not detected by the algorithm. This often occurs for areas that are very small relative to the surrounding tissue or that are unusually bright, resembling the background (Figure 3A). In some cases, these undetected regions are not minor; for example, in Figure 3B, the algorithm failed to accurately identify tissue because the area was particularly bright. Such cases highlight the limitations of the segmentation algorithm in handling low-contrast or atypically colored tissue regions.

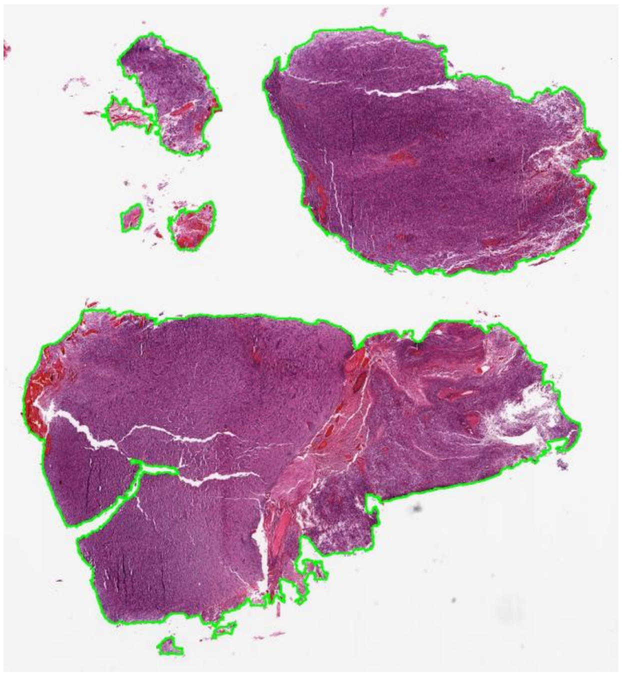
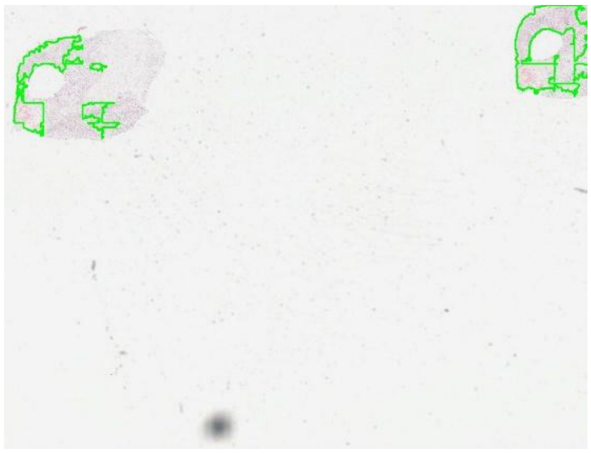
(A)  (B) 

Figure 3. Examples of tissue regions missed by the algorithm, including small or bright areas.

2.3 Are there any other artifacts on the slide (e.g., a signature of the slide company like “snowcat”, pen marks)? If yes, are these artifacts mistakenly detected as tissue?

In some cases, the 'hest' algorithm successfully ignored pen marks or other slide artifacts, correctly classifying them as background. In other cases, however, these artifacts were mistakenly detected as tissue. Examples of both scenarios can be seen in Figure 4.

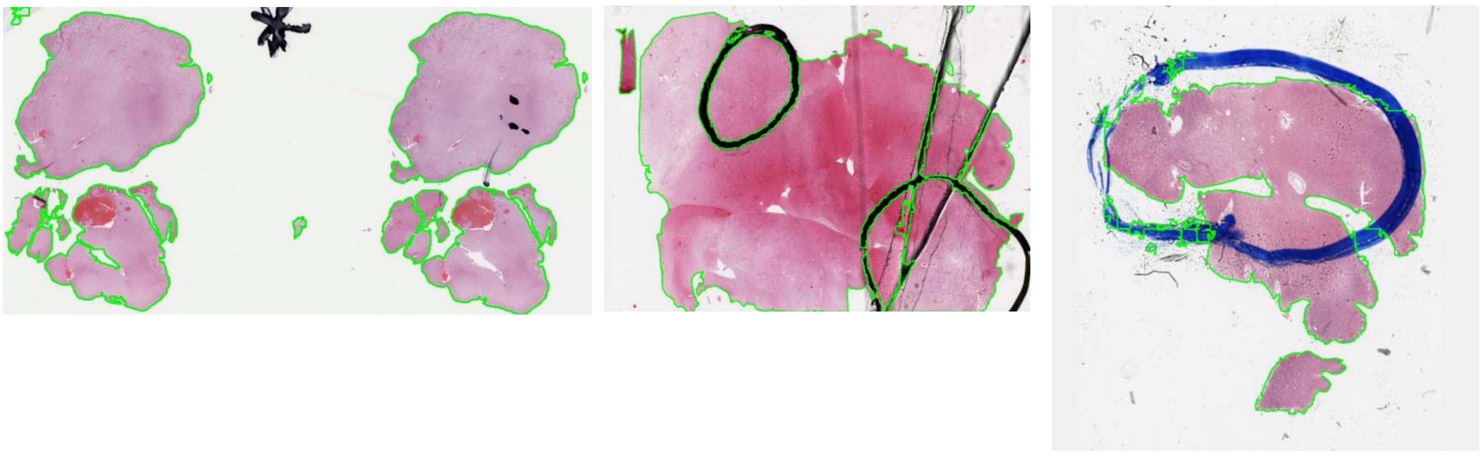


Figure 4. Examples of slide artifacts (pen marks), that were either correctly ignored or mistakenly classified as tissue.

3. Please add any other analysis that you think will be interesting to do.

An additional analysis that could be interesting is to evaluate the effect of applying a preprocessing step to the WSIs before segmentation. The current algorithm sometimes struggles with edge cases, for example when the tissue appears too light or when there are additional artifacts such as pen marks that can be mistakenly classified as tissue. A preprocessing stage (e.g., color- or threshold-based filtering such as Otsu [1]) could be applied to remove these artifacts and better separate tissue from background. This would allow the segmentation model to operate on cleaner inputs and potentially improve its robustness and accuracy.

Patch extraction

4.1. What patch size and magnification level did you use? Why?

I looked at the WSI sizes and found that the WSIs were scanned at different sizes (at level 0):

MPP range: 0.2456–0.5015 µm/pixel.

MPP (Microns per Pixel): MPP indicates the physical size each pixel represents in microns. For example, an MPP of 0.25 µm/pixel corresponds to a 40x scan, while 0.5 µm/pixel corresponds to 20x. Variations in MPP mean that patches of the same pixel size cover different real-world areas across slides.

Because of this range (20-40x), in order to maintain consistency across all WSIs, I chose mag 20 and patch size of 256x256 pixels, which is a commonly used in many studies [1], [2], [3], [4]. It provides a good balance between spatial detail and computational efficiency, allowing patches to capture sufficient tissue context while keeping the total number of patches manageable. Using mag 20 ensures each patch represents a consistent physical area, crucial for reliable deep learning. It also allows working with WSIs at lower resolution without requiring large up sampling, which could introduce blurring. Patch size represents a trade-off between coverage and computational cost: smaller patches provide finer spatial information and can potentially improve model performance, but they also increase computational load and memory usage. A patch size of 256×256 strikes a practical balance, giving sufficient tissue context while keeping processing feasible.

4.2. Display some of the output patches

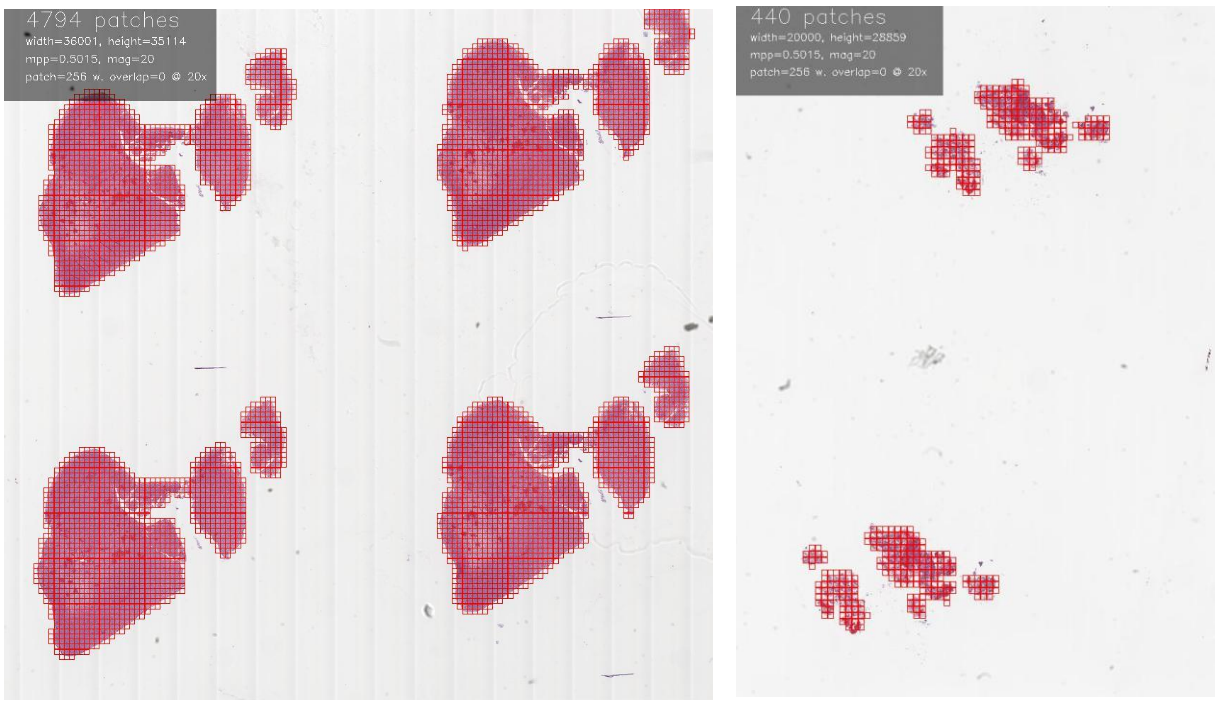

4794 patches
width=36001, height=35114
mpp=0.5015, mag=20
patch=256 w. overlap=0 @ 20x
440 patches
width=20000, height=28859
mpp=0.5015, mag=20
patch=256 w. overlap=0 @ 20x

Figure 5. Examples of selected 256×256 patches from WSIs at 20x magnification

4.3. Conduct data analysis, in particular: 4.3.1. Count how many patches were output per WSI and per case.

```
Average patches per WSI: 10432
Range of patches per WSI: 315 - 31262

Patches per WSI by IDH:
DHI False: mean=8431, range=386-24663
DHI True: mean=12354, range=315-31262
```

5. Please add any other analysis that you think will be interesting to do.

The average number of patched per WSI is 10432, and it range between 315-31262.

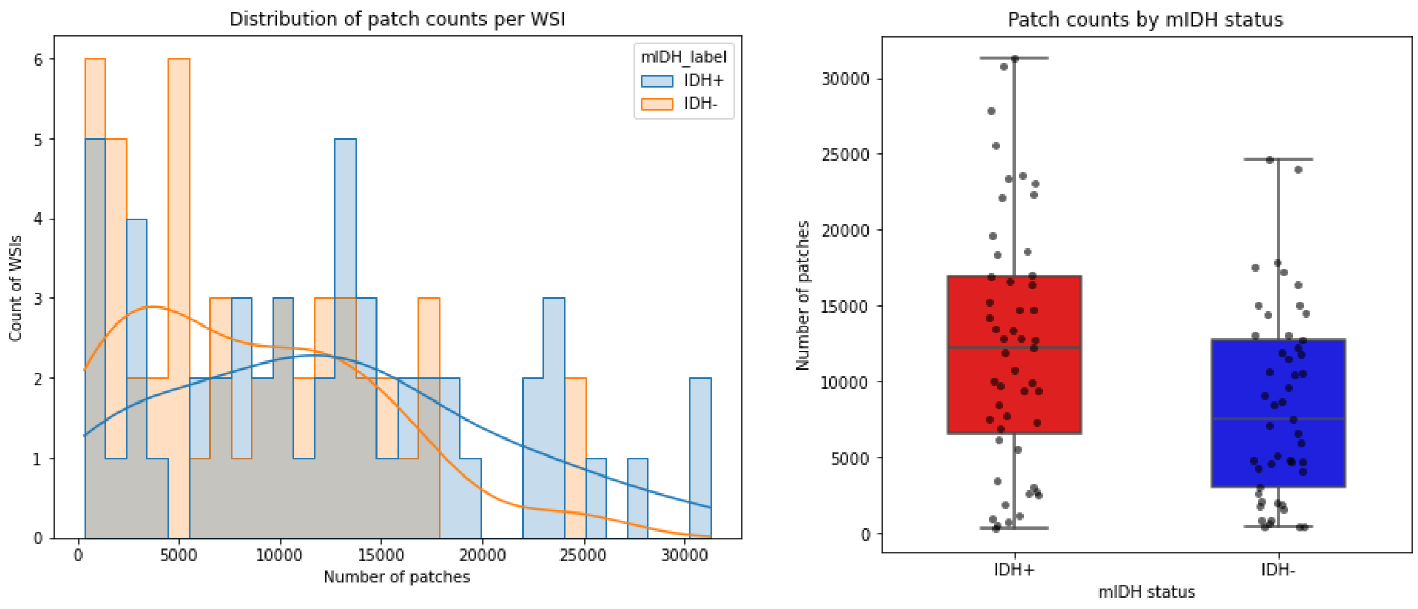


Figure 6. Distribution of patch counts per WSI and comparison between IDH mutation groups

WSIs from IDH+ cases generally contain a higher number of patches compared to IDH− cases, although both groups exhibit substantial variability and overlap. In both IDH+ and IDH− cohorts, patch counts range from several hundred to tens of thousands per slide, reflecting large differences in tissue size across WSIs.

Embeddings extraction

Choose one of the foundation models suggested in Trident, and extract embedding from the patches (task=feats). 6.1. What were the reasons behind choosing the model you chose?

For the feature extraction stage, I chose to focus on UNI2, as it is a self-supervised model trained specifically on large-scale pathology datasets. UNI2 is designed to capture morphologically relevant features in histopathology, such as nuclear patterns, vascular structures, and tumor architecture. A key advantage of UNI2 is that it produces embeddings at the patch level rather than only at the slide level. In the context of IDH mutation prediction, patch-level feature extraction is critical because not every region of a slide contains relevant morphological signatures, and their presence in certain areas does not necessarily indicate the status of the whole slide. By working at the patch level, an attention-based MIL framework can learn to identify which patches carry meaningful signals and assign them greater importance in the final prediction. UNI2 also generates compact 1536-dimensional vectors, balancing informativeness with computational efficiency and avoiding extensive fine-tuning.

As a comparison, I also include ResNet50, a widely used feature extractor in IDH prediction studies [5]. Its main advantage is higher computational efficiency, while producing 1024-dimensional feature vectors. Although it is less specifically tailored to pathology compared to UNI2, its extensive use in past research provides a clear benchmark against which to evaluate the performance of more advanced models.

7. Use t-SNE to plot the leading two components of the extracted embeddings. Each point should represent a WSI and should be colored by the WSI-level label.

t-SNE is a dimensionality reduction algorithm that projects high-dimensional data into 2D while aiming to preserve the relative similarity between samples. Visualizing embeddings with a t-SNE plot provides an initial assessment of their quality before training a machine learning model, as WSIs with similar features are expected to appear close together in the plot. The t-SNE plot (Figure 7) shows partial separation between IDH-mutant and IDH-wildtype WSIs, but with considerable overlap. This limited separation likely reflects the averaging of patch embeddings, which also includes patches without the mutation signal, thereby diluting the discriminative information.

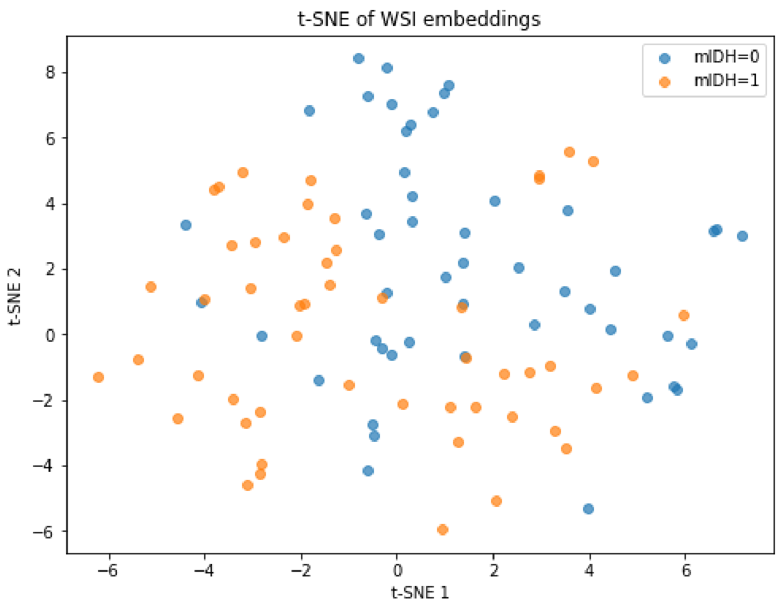


Figure 7. t-SNE for the patch embedding (UNI2)

IDH classification

8. Create a classifier for IDH mutation status. The classifier should take WSI-level or patch-level embeddings as input, and outputs a binary classification.

An SVM classifier was trained on the slide-level embeddings after average across UNI2 patch embedding. Then, I performed attention-based MIL model for patch embeddings.

I split the training set into 80% for training and 20% for validation. The validation set was used for hyperparameter tuning and early stopping: training was terminated when the validation loss did not improve for 3 consecutive epochs by at least 0.001, to prevent overfitting. The best-performing model, corresponding to the lowest validation loss prior to this stagnation, was saved. This approach ensures that the model stops training once it ceases to generalize, avoiding overfitting (Figure 8).

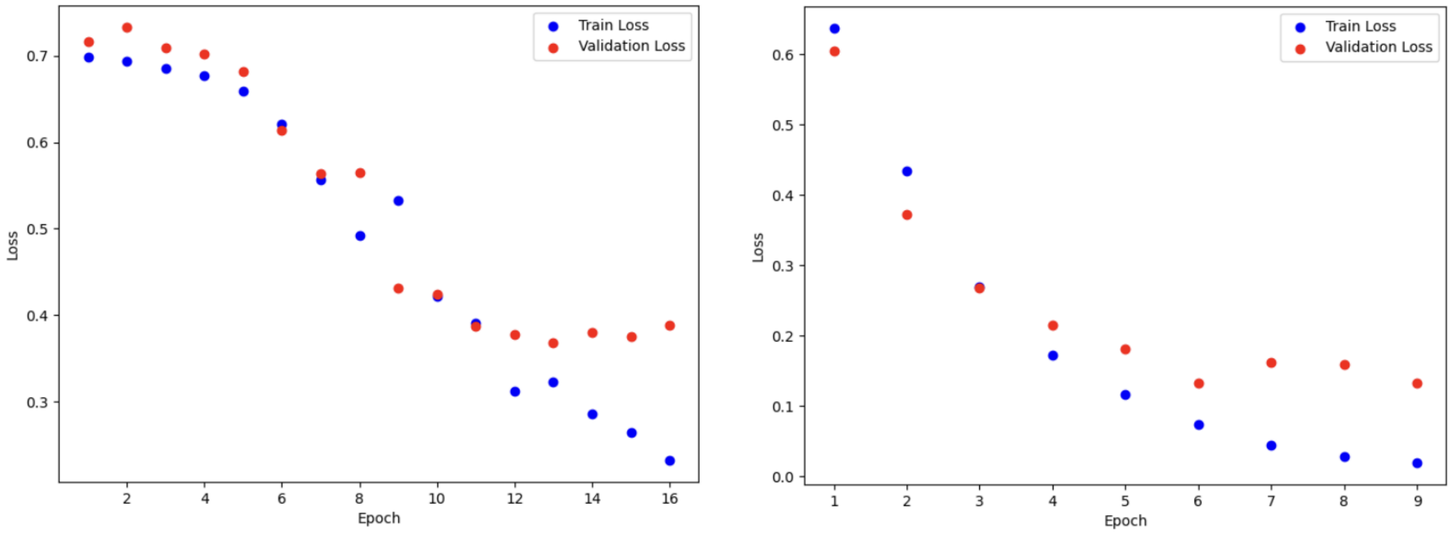


Figure 8. Training and validation loss per epoch (left: Resnet50 embedding; right: UNI2 embedding)

The plot shows that both Train Loss and Validation Loss decrease with the number of epochs, indicating that the model is learning and improving performance. However, a gap begins to appear between them, suggesting early signs of overfitting. Training is stopped after 3 epochs without significant improvement to prevent this.

9. Train the classifier on the train-set. 9.1. Which classifier did you choose and why?

The classifier consists of:

1. Attention-based MIL model: aggregates patch-level embeddings into a slide-level representation, giving higher weight to the most informative patches. Architecture:

- Attention module: two fully connected layers with a hidden size of 128 and a Tanh activation in between [6].
- Classifier module: two fully connected layers with a hidden size of 128, ReLU activation, followed by a sigmoid output for binary classification.

Due to the small dataset size, a deeper or more complex model was avoided to prevent overfitting.

2. Adam optimizer: adaptive optimization algorithm that adjusts the learning rate for each parameter individually, enabling efficient and stable training.

3. L2 regularization (weight decay): penalizes large weights to reduce overfitting.

4. Early stopping: monitors validation loss and stops training when performance ceases to improve for at least 0.001 for 3 epochs, preventing overfitting.

9.2. Report the classification metrics and assess the classifier performance.

Typically, model performance is evaluated on a separate test set that is kept aside throughout training and not exposed to the model until the evaluation stage. Since ground-truth labels for the designated test set are not available, I evaluated the model's performance on the labeled dataset instead.

classifier performance on all the train set (100 WSI)	Accuracy	Precision	Recall	F-score	AUC
Mean on patch UNI2 embedding + SVM	0.9600	0.9608	0.9608	0.9608	0.9600
UNI2 embedding + Attention-based MIL model	0.9900	0.9808	1.000	0.9903	1.0000
Resnet50 embedding + Attention-based MIL model	0.8900	0.8571	0.9412	0.8972	0.9504

All models perform well on the entire training set, with the UNI2 embedding (without SVM) clearly leading. I now evaluate performance on the validation set, which the models were not trained on, since training performance alone does not indicate generalization ability.

classifier performance on the validation set (20 WSI)	Accuracy	Precision	Recall	F-score	AUC
Mean on patch UNI2 embedding + SVM	0.9000	1.0000	0.7500	0.8571	0.8570
UNI2 embedding + Attention-based MIL model	0.9500	0.8750	1.0000	0.9333	0.9615
Resnet50 embedding + Attention-based MIL model	0.9000	0.8750	1.0000	0.9333	0.8333

The table shows that all models perform well on the validation set, with an Accuracy of at least 0.9. The UNI2 embedding with attention-based MIL model achieved the best performance.

9.3. You might get that the performance on the train-set is very high. What might be the reasons for that? And how would you check your assumptions? (only explain).

When the model achieves high performance on the training data, it indicates that it can learn from the data and perform classification well. This does not necessarily indicate a problem, as long as the model can also generalize to unseen data (evaluated on a test set), which would demonstrate that the model classifies effectively. However, when the model does not perform well on new (unseen) data, this can arise from several factors:

- Overfitting: the model learns the training data too well and fails to generalize to new data. This usually occurs when there is an imbalance between model complexity and dataset size: a very deep and complex model trained on a small dataset can memorize the existing data but cannot handle new samples. Additionally, it is important to consider the distribution in feature space: if the model has not been exposed to all possible variations in the data, it may "capture" only a small subset of the existing patterns, increasing the risk of overfitting and reducing generalization capability.
- Class imbalance: when the number of examples per category differs significantly, the model may bias predictions toward the dominant class, resulting in poor performance on minority classes.

How to check these assumptions:

- Evaluate on a separate test set: compare training vs. validation performance, a large gap indicates overfitting.
- Analyze class distribution: verify that all classes are represented reasonably and that high performance isn't just due to predicting the majority class.
- Inspect learning curves: plot loss and accuracy over epochs for both training and validation sets. Divergence between them signals overfitting; for example, if training performance improves but validation performance does not, the model is likely overfitting.

10. Please add any other analysis that you think will be interesting to do.

Previous studies have shown that incorporating patient clinical information can improve the predictive performance of the algorithm [7]. Therefore, another potential approach is to integrate clinical data and use an ensemble of the two models to achieve better classification results.

Reference

[1] A. Zhang, G. Jaume, A. Vaidya, T. Ding, and F. Mahmood, “Accelerating Data Processing and Benchmarking of AI Models for Pathology,” Feb. 10, 2025, *arXiv*: arXiv:2502.06750. doi: 10.48550/arXiv.2502.06750.

[2] “Artificial Intelligence for Digital and Computational Pathology.” Accessed: Sep. 10, 2025. [Online]. Available: https://arxiv.org/html/2401.06148v1/#S2

[3] Z. Fu, Q. Chen, M. Wang, and C. Huang, “Whole slide images classification model based on self-learning sampling,” *Biomed. Signal Process. Control*, vol. 90, p. 105826, Apr. 2024, doi: 10.1016/j.bspc.2023.105826.

[4] N. Coudray *et al.*, “Classification and mutation prediction from non–small cell lung cancer histopathology images using deep learning,” *Nat. Med.*, vol. 24, no. 10, pp. 1559–1567, Oct. 2018, doi: 10.1038/s41591-018-0177-5.

[5] Q. Lv, Y. Liu, Y. Sun, and M. Wu, “Insight into deep learning for glioma IDH medical image analysis: A systematic review,” *Medicine (Baltimore)*, vol. 103, no. 7, p. e37150, Feb. 2024, doi: 10.1097/MD.0000000000037150.

[6] M. Ilse, J. M. Tomczak, and M. Welling, “Attention-based Deep Multiple Instance Learning,” Jun. 28, 2018, *arXiv*: arXiv:1802.04712. doi: 10.48550/arXiv.1802.04712.

[7] R. Nakagaki, S. S. Debsarkar, H. Kawanaka, B. J. Aronow, and V. B. S. Prasath, “Deep learning-based IDH1 gene mutation prediction using histopathological imaging and clinical data,” *Comput. Biol. Med.*, vol. 179, p. 108902, Sep. 2024, doi: 10.1016/j.compbiomed.2024.108902.