Project Machine Learning
— Milestone 3 —

Filip Matysik, usr: pml16

February 7, 2025

GitHub: `https://github.com/filippuo2000/TransMIL_PML_TUB_WS24_25`

1 Introduction

This report describes the work that has been done for milestone 3 of the project that is aiming to replicate the results of the TransMIL (Shao et al. (2021)) paper, specifically on the CAMELYON16 (Ehteshami Bejnordi et al. (2017)) dataset.

The main goal of this milestone was to perform the final model selection (the one performed in the Milestone 2 lacked reproducibility - all models have been ran only once, with the very same set of initial weights) and apply the xAI (explainable AI) methods on the selected model, to understand its decision making process.

For final model selection, to assure fair and meaningful comparison, the best four performing models from the milestone 2 have been trained in 10 runs each, with different initial weights in each run.

For xAI part, three different methods have been implemented - Gradient Saliency Maps, Integrated Gradients and Attention Rollout. For the two of them, both the heatmaps and quantitative scores summarising the model's behavior have been obtained.

2 Final model selection

In order to choose the best performing model, 4 different models have been compared. For simplicity, those were the models that achieved the best results in the Milestone 2. Normally one should choose the models for a comparison from a wider range of parameters, especially since the models in milestone 2 were only trained and evaluated on a single set of initial weights, thus one could argue that they obtained better results than the other models only due to the fact that the given set of initial weights was favoring them. However, due to the limited amount of time and human resources, the model selection process has been simplified and limited to those 4, pre-chosen models, although it has been conducted taking into consideration the best model selection practices.

Configuration of the models selected for the final comparison:

- **Model 1**: 512 input features (each patch downscaled with a fully connected layer), **without** the PPEG module, SGD optimizer.
- **Model 2**: 768 raw input features (each patch processed without a fully connected layer), **with** the PPEG module, SGD optimizer.
- **Model 3**: 384 input features (each patch downscaled with a fully connected layer), **with** the PPEG module, SGD optimizer.
- **Model 4**: 384 input features (each patch downscaled with a fully connected layer), **with** the PPEG module, Lookahead+RAdam optimizer.

All models were trained with a learning rate of 0.002, for 30 epochs, with early stopping callback and its patience parameter set to 5 epochs, with a batch size of 1. Due to the varying sequence lengths of whole-slide images (WSIs), training with a larger batch size would require modifying the model's sequence squaring operation. Specifically, one would need to apply consistent padding across all input sequences so that the length of all sequences after padding matches the longest padded sequence in the batch.

To guarantee a fair comparison, all versions of the models were trained 10 times, every time with a different set of initial weights. The initial weights were also different among runs for every model. In order to select the best performing model, best scores for different metrics from every run were selected. This selection only considered metric scores obtained after 5 epochs of training - having no learning rate scheduler and no learning rate warmup, sometimes a model would hit a very high score for a given metrics at the very early stage of training, which is not desired for such a complex model, thus those scores should be rejected. For final model selection, the following 3 different metrics have been taken into consideration - positive (1) class accuracy (Recall), AUROC score, overall Accuracy. The comparison has been presented on figures 1., 2. and 3.

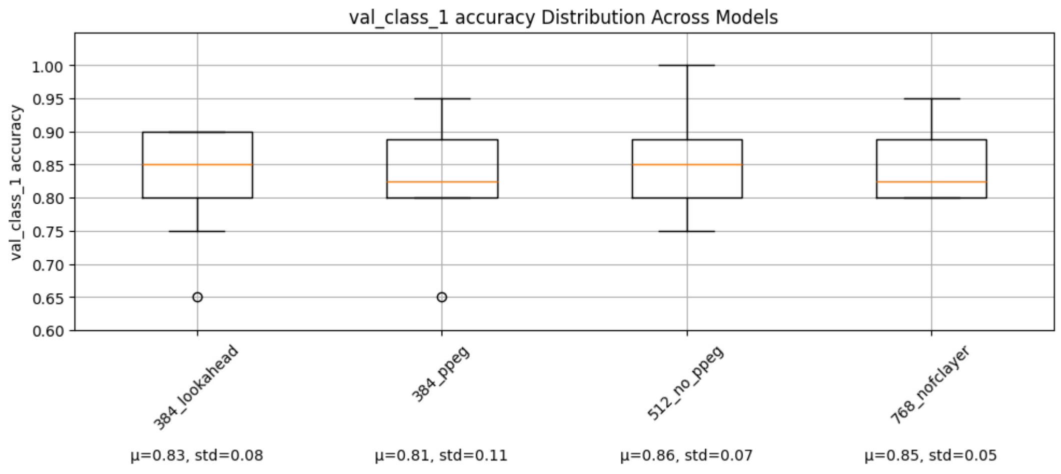


Figure 1: Comparison of different models for class 1 accuracy (recall) metrics

As can be seen on figure 1., the 512_no_ppeg model achieves both the highest median and mean score among all models. Even though its standard deviation is slighty higher than the one for the 768_nofclayer model, that difference should not be considered significant, especially since the 512_no_ppeg model simply achieves higher maximum and lower minimum scores than the 768_nofclayer. At this point, those two models should be considered best for this specific metrics. The remaining two have a smaller mean and they both obtained unacceptably low scores for this metrics in one run out of ten - they thus have a larger standard deviation and seem less reliable.

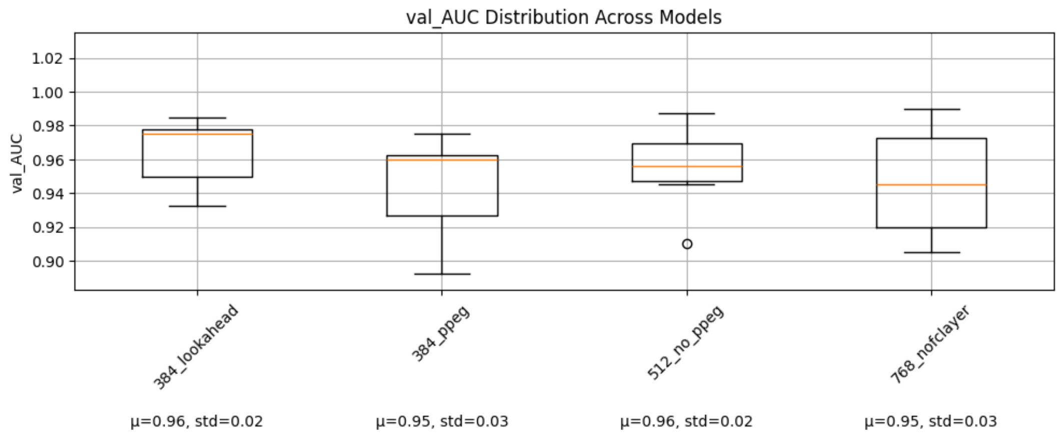


Figure 2: Comparison of different models for AUROC metrics

When moving to figure 2. and the AUROC score, it again can be seen that the 512_no_ppeg model together with 384_lookahead has the highest mean and smallest standard deviation. Even though most AUC scores were higher for the 384_lookahead, it has been shown earlier that this model achieved worse results than 512_no_ppeg on the class 1 accuracy metrics, thus for those two having similar results on the AUC metrics, one should still pick the 512_no_ppeg model, since the class 1 accuracy should be considered a more important metrics for the given problem (Cancer classification). The 768_nofclayer model, that performed similarly well to the 512_no_ppeg on the class 1 accuracy, for AUROC has a slightly higher variance and lower mean and median values than 512_no_ppeg, thus out of the two, one should select the latter. This is indeed confirmed in figure 3., where accuracy scores are presented.

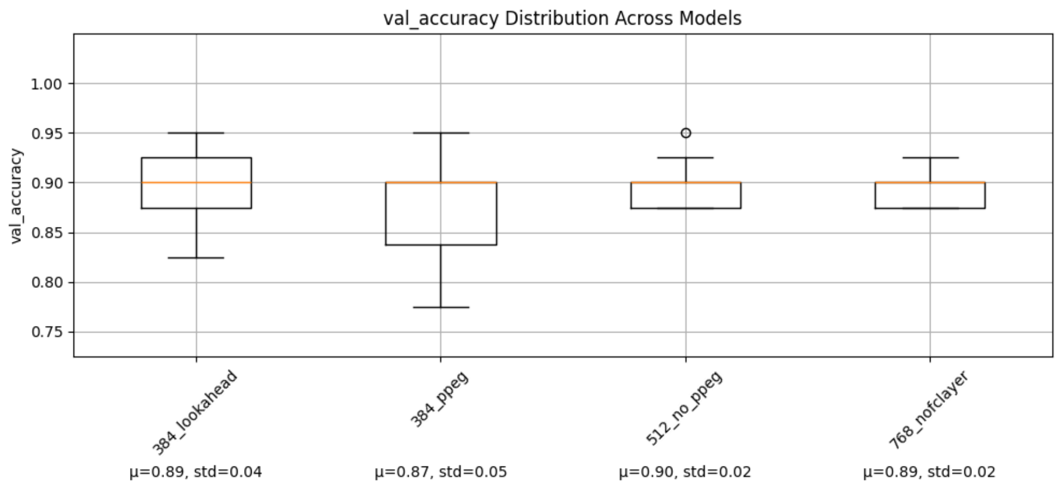


Figure 3: Comparison of different models for accuracy metrics

For accuracy metrics (figure 3.), the 512_no_ppeg has the highest mean out of all models and together with the 768_nofclayer model, the smallest standard deviation. Considering that for two previously analyzed metrics, this model has performed best, it is chosen as the final model out of 4. Another step is then to select the best run for this very model in order to obtain the results on the test set.

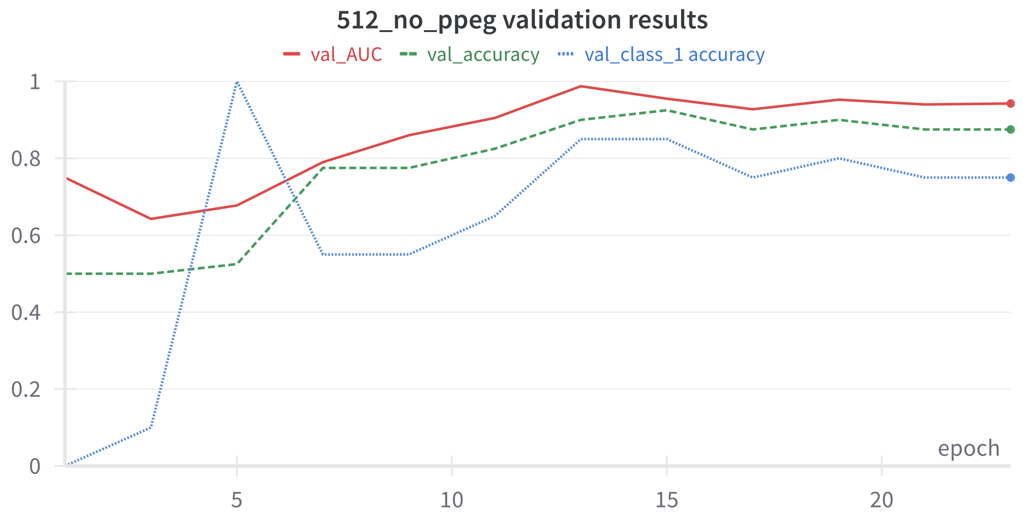


Figure 4: Validation results for the best model

In figure 4., validation scores for previously analyzed metrics of the best run of the selected final 512_no_ppeg model, have been presented - best checkpoint has been saved for epoch 13, based on the minimum validation loss. The best run has been selected in a conceptually similar way to the best model selection. Firstly, results for the class 1 accuracy (Recall) were analyzed and then for all the possible candidates, one also had to check whether for a given run the model is not trading off AUROC score and Specificity and Accuracy scores for class 1 accuracy score. This would be undesired. The best run can then be seen as the most stable one, in which at the same time high scores are achieved for the class 1 accuracy and for the remaining metrics, having also a stable validation loss scores, without exploding values past the epoch with the minimum validation loss function value.

Below table presents the results of the inference on the test set for the best model and its best run, in comparison with the original TransMIL model's results:

Table 1: Test results comparison

Name	**Recall (%)**	**AUC (%)**	**Acc (%)**	**Specificity (%)**
512_no_ppeg_larger_lr	89.8	**95.00**	**91.47**	92.50
TransMIL paper model	n/a	93.09	88.37	n/a

The chosen model clearly outperforms the original TransMIL model for both Accuracy and AUROC metrics and at the same time achieves a satisfying 89.8 % score for the Recall (class 1 accuracy).

3 xAI - explainability

To further analyze the model's performance, 3 different xAI methods have been implemented. They were utilized to generate the per-patch importance scores, which allowed for the generation of visual heatmaps presenting where the model focuses its attention in the classification process. Besides the heatmap, a quantitative summary of the model's decision process has also been obtained.

All experiments in this part have been ran on the samples from the CAMELYON16 test set, with the model set to evaluation mode. Analyzed model was simply the one chosen in the model selection part - 512_no_ppeg and its best run.

For all three methods, the final per-patch scores were squeezed into 0-1 range. If the sample's label is 1, ideally all patches that contain the cancer regions should aim for that score and when the label is 0 (no cancer regions), all labels should aim for the lowest possible score.

3.1 Gradient Saliency Maps

The simplest method out of the implemented is the Gradient Saliency Maps method (Simonyan et al. (2014)). Here, for a given input sample, after computing the forward pass, one should simply calculate the gradient of the output (raw logits) for the higher value out of the two logits with respect to the input, thus with respect to the sequence of patches and their individual features. After those gradients have been obtained, one has to take an absolute value for all of them and sum them across the feature dimension for each patch in the input sequence (sample). If it wasn't for taking the absolute values, gradients would cancel each other out, leading to a loss of information. These gradient values are then normalized by subtracting the minimum gradient from all elements and dividing the difference by the difference between the maximum and minimum gradient in the sequence. This way the values are squeezed into 0-1 range and do not form a probability distribution.

Squaring of the gradients has been substituted with taking the absolute value, because squaring the gradients was making most of them almost uniform and around the 0 value after normalization, thus decreasing the possibility to reasonably analyze the input of given patches, which was not the case when the absolute value was extracted instead. After all, the higher the input gradient for a given patch, the higher this patch influences the output classification result towards the positive label. Out of three, this method is the least computationally expensive.

3.2 Integrated gradients

The Integrated Gradients method (Sundararajan et al. (2017)) is a bit more complex than the Gradient Saliency Maps. Here authors of the paper argue that a better way to analyze the importance of input patches (originally pixels or image regions) would be to filter out the important input parts by calculating the gradients w.r.t input sample for modified versions of this very input, along a straight path, starting from the so called "baseline" input (which in the image case would be a black image, that could be considered to have a "neutral" label and for the case of feature vectors, the baseline would be a vector with all its values set to 0) and then moving towards the original input in the subsequent steps. Essentially instead of computing gradients at a single point, Integrated Gradients estimates the input regions' importance by accumulating gradients along that path. The idea here is to analyze for which input features, modifying them changes the output of the model - such input features thus influence the model's performance the most.

For Integrated Gradients a good practice would be to compute around 50 steps for each input sample before returing the input attributions (importance scores). However due to the large size of the input samples in the CAMELYON16 dataset, this has proven to be nearly impossible to compute. After trying to increase the available RAM for a given run, even allocating 32x4 GB of RAM was not sufficient to deal with the longest sequences from the test dataset. This was the case also after reducing the number of required steps to 10, which should already be considered as not enough for this method, especially when dealing with such complex input data. Thus no final results for this method have been obtained.

3.3 Attention rollout

Finally the Attention Rollout method Abnar and Zuidema (2020) analyzes specifically the Transformer layers of the model. In deep transformers, the attention at the final layer alone does not provide a clear explanation of how much each input token (or patch) contributes to the final decision. To tackle this problem, in Attention Rollout one has to recursively multiply the attention matrices for subsequent layers in the model, starting from the first layer. To make up for the residual connections (which are normally taken into account in the transformer layers), one has to add an Identity matrix to the attention matrix from the next layer (Identity and Attention matrices are both weighted by 0.5).

When using the Multi-Head Attention, after obtaining the rollout results for a given attention layer (which are used for the explainability and patch-level importance score assignment), one has to take the mean of the attention weights across the head dimension (mean is suggested by the authors of the method). After that each row in the attention matrix is divided by the sum of the elements in this row, which results in a probability distribution for each row. This is done before the matrix multiplication with the previously collected results. At the final layer, one has to take out the part of the first row of the computed result attention matrix, which contains the information about the total attention between the class token and all patches in the sequence. That's how the importance scores for each patch are thus obtained (after being normalized to the range of 0-1 in the end).

3.4 xAI - quantitative results

To obtain the quantitative summary of how certain parts of the model are responsible for classification, ROC curve and AUROC score have been computed for the whole test set and given xAI methods outputs. This imposed the need to obtain the patch-level labels for all the test samples. If a patch contains any cancer regions - it is assigned a ground truth of 1. Otherwise, it is assigned a ground truth score of 0. Thus patches in the negative samples from the test set are all marked with the ground truth score of zero. Predictions for every patch are simply the scores that were obtained through the two xAI methods: Attention Rollout and Gradient Saliency Maps. To compute ROC curve and AUROC, predicitions and labels have been concatenated for all the samples in the test set. ROC curve is presented below on the figure 5.:

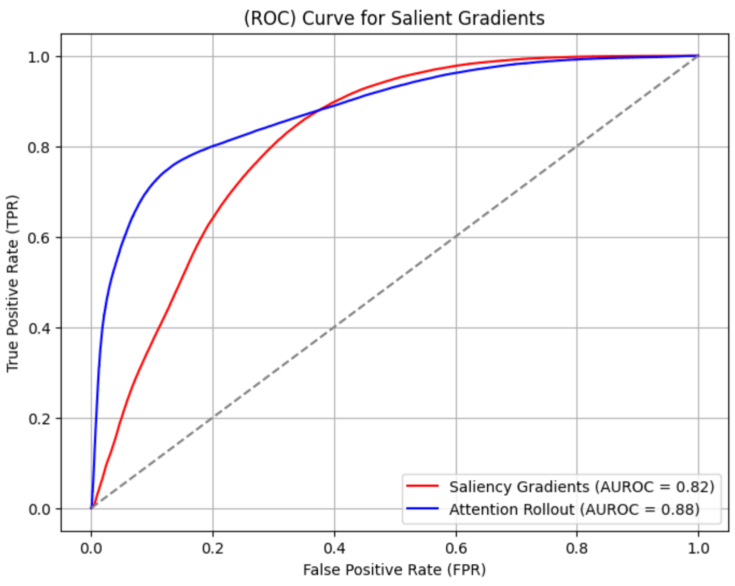


Figure 5: ROC curve for Attention Rollout and Gradient Saliency Maps methods

As can be seen in figure 5., the difference between the positive and negative patches is better captured by the attention layers than by the gradients of the output with respect to the input. This only proves that when analyzing the model's decision making process, one should primarily look at the transformer layers explainability - this provides more meaningful insight into how the model makes its predictions.

3.5 xAI - visualization

In this section some heatmaps have been visualized for the visual interpretation of the model's decision making process. Since the Attention Rollout method has proven to be more efficient than the Gradient Saliency Maps method, maps have been obtained using the Attention Rollout.

3.5.1 Positive sample - visualization

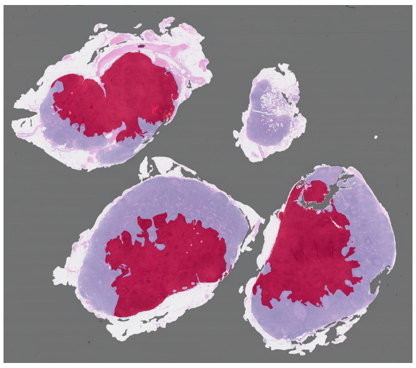
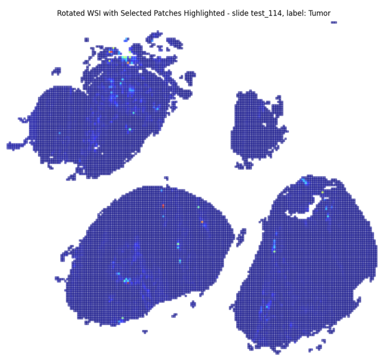


Figure 6: Annotated positive sample slide


Rotated WSI with Selected Patches Highlighted - slide test_114, label: Tumor
Figure 7: Heatmap for the positive sample slide

In the above figures 6. and 7. respectively it can be seen that the model assign its weights to the correct parts of the positive sample (to where the cancer regions are). Although this attention is not as dense and spread across a wider region, but rather focuses in a few points that belong to the cancer regions.

3.5.2 Negative sample - visualization

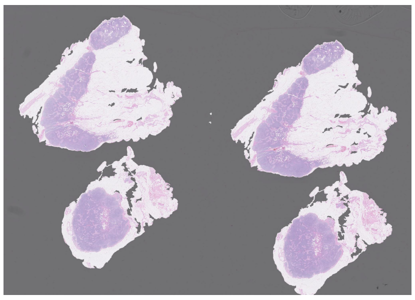
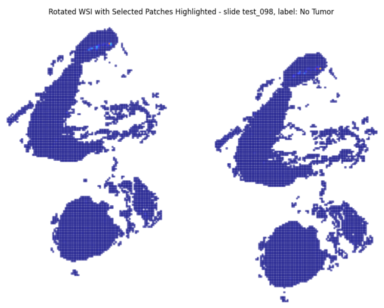


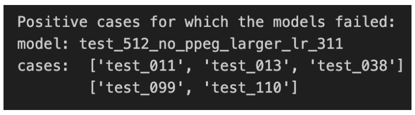
Figure 8: Negative sample slide


Rotated WSI with Selected Patches Highlighted - slide test_098, label: No Tumor
Figure 9: Heatmap for the negative sample slide

For a negative sample, the output of the heatmap is also as expected. Here the model assigns very little attention to any regions in the slide, with only two exceptional regions (however importance scores there aren't high).

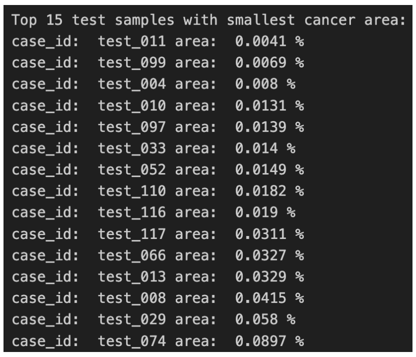
3.5.3 False negative errors - visualization

Below, on figures 10. and 11. a brief analysis of the worst cases has been performed (incorrectly classified positive cases - false negative errors). It can be seen that the model fails on the 'test_011' and 'test_099' samples, which turn out to have the smallest percentage of the cancer area in the whole test set, thus these could be considered the two most "difficult" cases. In figure 13. a heatmap obtained with Attention Rollout is presented for the 'test_099' sample:



```
Positive cases for which the models failed:
model: test_512_no_ppeg_larger_lr_311
cases:  ['test_011', 'test_013', 'test_038']
        ['test_099', 'test_110']
```

Figure 10: Uncorrectly classified positive cases



```
Top 15 test samples with smallest cancer area:
case_id:  test_011 area:  0.0041 %
case_id:  test_099 area:  0.0069 %
case_id:  test_004 area:  0.008 %
case_id:  test_010 area:  0.0131 %
case_id:  test_097 area:  0.0139 %
case_id:  test_033 area:  0.014 %
case_id:  test_052 area:  0.0149 %
case_id:  test_110 area:  0.0182 %
case_id:  test_116 area:  0.019 %
case_id:  test_117 area:  0.0311 %
case_id:  test_066 area:  0.0327 %
case_id:  test_013 area:  0.0329 %
case_id:  test_008 area:  0.0415 %
case_id:  test_029 area:  0.058 %
case_id:  test_074 area:  0.0897 %
```

Figure 11: Top 15 test cases with the smallest cancer region percentage

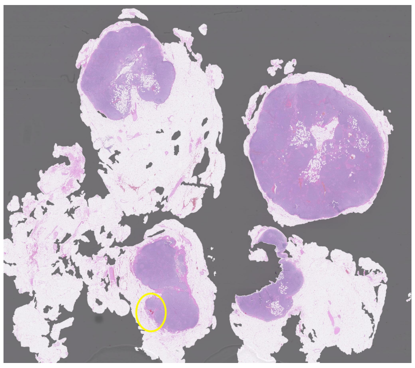
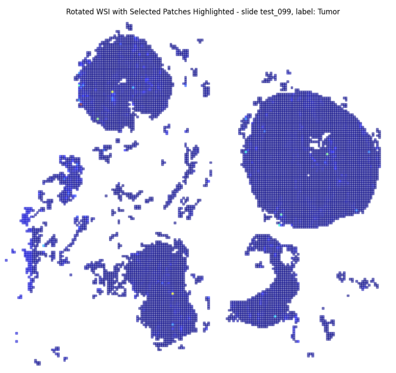


Figure 12: False Negative small cancer area annotated slide


Rotated WSI with Selected Patches Highlighted - slide test_099, label: Tumor
Figure 13: Heatmap for the False Negative small cancer area annotated slide

On figure 12. the cancer region has been additionaly marked with the yellow circle for visibility, as it is difficult to capture. On figure 13. it can be seen that the model misses this region and incorrectly assigns its attention (however little) to different regions.

4 Discussion

Milestone 3 finalizes the TransMIL model implementation. At this point, the final selected model outperforms the model provided by the authors of the TransMIL paper, thus in someway the main goal of this project has been achieved.

Implemented xAI methods provide meaningful insight into model's decision making and show that the model is actually paying attention to the regions that contain cancer cells when making the classification decision (which is desired). xAI methods that analyze the attention mechanism turn out to be more meaningful for the explanation of the model behavior. High AUROC score for the Attention Rollout proves that even on the patch level, the trained model is good at distinguishing between the positive and negative samples. One could further investigate to what extent such a TransMIL model, trained in the weakly supervised way with slide level labels only, could be utilised to not only perform classification of the samples, but also act as a segmentation method (which would be equivalent to extracting the cancer regions from patches). Such an idea is mentioned in the section 3.2 of the Simonyan et al. (2014) paper.

There's still room for improvement in the way heatmaps are generated - the current method is able to grasp and visualize the general behavior of the model in the correct manner, but not the cleanest possible.

References

S. Abnar and W. Zuidema. Quantifying attention flow in transformers, 2020. URL `https://arxiv.org/abs/2005.00928`.

B. Ehteshami Bejnordi, M. Veta, P. Johannes van Diest, B. van Ginneken, N. Karssemeijer, G. Litjens, J. A. W. M. van der Laak, , and the CAMELYON16 Consortium. Diagnostic assessment of deep learning algorithms for detection of lymph node metastases in women with breast cancer. *JAMA*, 318(22):2199–2210, 12 2017. ISSN 0098-7484. doi: 10.1001/jama.2017.14585. URL `https://doi.org/10.1001/jama.2017.14585`.

Z. Shao, H. Bian, Y. Chen, Y. Wang, J. Zhang, X. Ji, and Y. Zhang. Transmil: Transformer based correlated multiple instance learning for whole slide image classication. *CoRR*, abs/2106.00908, 2021. URL `https://arxiv.org/abs/2106.00908`.

K. Simonyan, A. Vedaldi, and A. Zisserman. Deep inside convolutional networks: Visualising image classification models and saliency maps, 2014. URL `https://arxiv.org/abs/1312.6034`.

M. Sundararajan, A. Taly, and Q. Yan. Axiomatic attribution for deep networks, 2017. URL `https://arxiv.org/abs/1703.01365`.