Presentation 4

- Links to presentation(s) and code(s) on GitHub
    - Presentation
    - Code for effective attention analysis
- What did you do?
    - Calculate effective attentions for all patches and wrote multiple functions:
        - compute_effective_patch_attention() function to calculate effective attention for all patches
            - effective attention: stain_attn_weight * slice_attn_weight * patch_attn_weight
        - plot_effective_patch_attention_distribution_per_case() function plot how effective attention weights are distributed (1 distribution plot per case)
        - analyze_top_effective_patches_per_case() function shows the "identity" of the top n patches (in terms of effective attention) for each case. "Identity" means which stain & slice are these top effectively attending patches belongs to.
            - The idea behind this is to see if the effective attention is only focusing on patches in only a few high attending slices, or is it taking top attending patches from various slices. And how that contributes to the prediction
        - visualize_case_effective_patches() function visualize the top and bottom n patches in terms of effective attention for each case
- How does it help the project?
    - Effective attention analysis can help us diagnose issues with the model.

      These codes are reusable for future models. Answering questions like:
        - Does it work better for the model to focus on only a few cases or on more cases?
        - Do the cases that got predicted correctly have high-attending patches from multiple slices, or patches from only a few high-attending slices?
- Issues faced (if any)
- Attempts to resolve issues (if any)

- Issues resolved (if any)
- Next steps
    - Hyperparameter tuning for the model
    - Run effective attention analysis on the cross-validated models to try to see problems
- References (Mention if you built up on someone else's work)
    - GenAI tools

Presentation 3

- Links to presentation(s) and code(s) on GitHub
    - Presentation
    - Code for different color space experimentation
    - Code for color normalization
    - Code for attention analysis
- What did you do?
    - Run Spring's code with HED color space without color normalization parameters on Google Colab (took ~3 days, this is before we got Quest) and benchmarked the result with RGB color space
    - Implement code in Google Colab on getting color normalization parameters for the HED color space using a sample of our training patches
    - Implement code in Quest use for switch color space to HED, LAB, HSV
    - Implement visualization for attention analysis with Paisley
- How does it help the project?
    - Help understand whether changing the color space would improve performance
    - Better visualization for attention (both high-grade and benign, high and low attention)
- Issues faced (if any)
    - Since only the RGB color space has a set of color normalization, experimenting with other color space for each stain would require manually getting the color normalization from our data, which could take a long time
- Attempts to resolve issues (if any)

    - Implemented code for that, will migrate the code into Quest and change into python file format
- Issues resolved (if any)
- Next steps
    - Migrate normalization code in Quest and get all parameters
    - Rewrite the transformation function so that each stain uses a different color space
    - Change the visualization for the attention analysis function so it shows the post-transformed images to give us a better understanding
- References (Mention if you built up on someone else's work)
    - GenAI tools

Presentation 2

- Links to presentation(s) and code(s) on GitHub
    - https://docs.google.com/presentation/d/19nkjkmNrdSh5d7TP2qkPXQKUFbikS9LrFrL17EElqhY/edit?slide=id.p#slide=id.p
    - Spring Model Rerun: https://github.com/arvindkrishna87/STAT390_CMIL_Fall2025/blob/main/Code/MIL/springMIL_rerun_22Oct_Luna.ipynb
    - Color Conversion (in progress)https://github.com/arvindkrishna87/STAT390_CMIL_Fall2025/blob/main/Code/MIL/colorbenchmark_22Oct_Luna.ipynb
- What did you do?
    - Coding
        - Run Hannah's model with H&E stains only (trained for ~1.5 days) with added data & correct train-test split.
        - Add a function to save checkpoints to Google Drive instead of the local path, as Hannah did: `save_checkpoint_to_drive(model, arch, optimizer, epoch, drive_folder="MyDrive/Checkpoints")`
        - Add function to load the latest checkpoint: `load_latest_checkpoint(checkpoint_dir, model, optimizer, device)`
        - Change the train_model function to train from last checkpoint: `train_model(model, optimizer, criterion, train_loader, val_loader, arch, epochs=5, start_epoch=0)`

        - Change the model eval logic so it actually saves attention weights! Original code rerun model eval multiple times for getting prediction for slice, patch and case & for getting attention weights to visualize.
        - Start coding the color conversion part to benchmark with Hannah's model.
    - Research
        - Research way to implement other color spaces, specially focusing on h&e. Read on paper relating to color decolonization: separating hematoxylin channel & eosin channel.
- How does it help the project?
    - Establish a baseline model evaluation (rerunning hannah's model) for later comparison
    - More sustainable code and helpful functions
- Issues faced (if any)
    - Training time is too long, making it hard to test ideas.
- Attempts to resolve issues (if any)
    - Will switch to quest, hopefully will make it faster
    - Just do 3 epoch for benchmarking instead of 5 epoch
- Issues resolved (if any)
- Next steps
    - Complete the 2 action plans mentioned above
    - Since we decided to use Option A, I can also try to tune color space independently for each stain. As mentioned, brightness would benefit H&E stain, but for IHG stains, color seems to be more important.
- References (Mention if you built up on someone else's work)

- https://scikit-image.org/docs/0.25.x/auto_examples/color_exposure/plot_ihc_color_separation.html

Presentation 1

- Links to presentation(s) and code(s) on GitHub
- What did you do?
    - I researched different methods of color representation and gained insights from past literature relating to histological imaging. I outlined 2 action plans: (1) transform RGB images into other well-performing color spaces, such as LAB and HSV, and benchmark with MIL model from last quarter; (2) test whether features containing brightness information are important for model training.
- How does it help the project?
    - Improving color representation could improve training data quality, leading to better classification. Brightness information is shown to be helpful in shape detection. By using a color space that decouples brightness from color, our model could potentially have better learning on the distribution of melanocytes.
    - The previous cross-stain model averaged out the RGB channels from three stains. If we found that the brightness channel alone performs well, we can use only the brightness channel for each stain, so that each stain has its own dimension, but the overall dimension is still kept low.
- Issues faced (if any)
- Attempts to resolve issues (if any)
- Issues resolved (if any)
- Next steps
    - Complete the 2 action plans mentioned above

    - Since we decided to use Option A, I can also try to tune color space independently for each stain. As mentioned, brightness would benefit H&E stain, but for IHG stains, color seems to be more important.
- References (Mention if you built up on someone else's work)
    - Madusanka, N., Jayalath, P., Fernando, D., Yasakethu, L., & Lee, B.-I. (2023). Impact of H&E Stain Normalization on Deep Learning Models in Cancer Image Classification: Performance, Complexity, and Trade-Offs. Cancers, 15(16), 4144. https://doi.org/10.3390/cancers15164144
    - Bishnoi, Vidhi, and Nidhi, Goel. (2023). A Color-Based Deep-Learning Approach for Tissue Slide Lung Cancer Classification. Biomedical Signal Processing and Control. www.sciencedirect.com/science/article/abs/pii/S1746809423005840