1. Who are you (mini-bio) and what do you do professionally? If you are on a team, please complete this block for each member of the team.

I am Tristan Lazard, and I am finishing my PhD in computational pathology in Mines-Paristech and Institut Curie under the supervision of Thomas Walter and Etienne Decencière. I worked on the prediction of *homologous recombination deficiency in breast cancer from histopathological images*, and on the technical side I focused on the different supervision regimes for deep learning models in the context of histopathological images (for example with *this work*, or *this one* )

2. What motivated you to compete in this challenge?

I aimed to test the algorithm I developed over the past year, which is a fully self-supervised method for training whole slide image (WSI) representations.

3. High level summary of your approach: what did you do and why?

I primarily followed the same approach as outlined in the paper, but with some adjustments:

1. Due to specific Docker installation issues, I could not use the sparse-CNN aggregator. Instead, I employed a spatially-unaware architecture consisting of residual fully connected blocks.
2. Given the high amount of normal dermal or epidermal tissue and background in the WSIs, I manually labeled a few patches to train a binary classifier for detecting them. This classifier was then used to filter out irrelevant patches for the task.

After encoding with the Giga-SSL pre-trained models, we obtained a 256-dimensional feature vector for each WSI. These L2-normalized embeddings were directly used to train L2-regularized logistic regression models in a 40-fold cross-validation scheme, which served as the classification models for the challenge.

4. Do you have any useful charts, graphs, or visualizations from the process?

5. Copy and paste the 3 most impactful parts of your code and explain what each does and how it helped your model.

```
TILE_ENCODER = "models/moco_tile.pth.tar"
```

The tile encoder bears a huge role in the prediction power of the model.

```
TRAIN_QUALITYCHECK()
```

This small tile-classifier used to filter tiles based on their embeddings is a key part of the model: the giga-ssl aggregation model is very sensitive to the proportion of key instances in the WSI.

```
X = encode_wsi(model, hook, wsi, resolution[wsi], device, out=outputs)
```

And of course the Giga-SSL encoding of the tiles, i.e. the self-supervised aggregation of the tiles embeddings into a single slide embedding. It allows to work with a single vector per WSI and ease a lot the investigation and experimentation done afterward.

6. Please provide the machine specs and time you used to run your model.

- CPU (model):
- GPU (model or N/A): V100
- Memory (GB): 16Gb
- OS: Linux
- Train duration: ~2h
- Inference duration: ~2h

7. Anything we should watch out for or be aware of in using your model (e.g. code quirks, memory requirements, numerical stability issues, etc.)?

If possible, use the Sparse-CNN aggregator rather than the MLP one provided here, as it is more efficient due to its incorporation of spatiality in the aggregation process. Additionally, it is crucial to L2-normalize the output of the Giga-SSL model before utilizing it for classification.

8. Did you use any tools for data preparation or exploratory data analysis that aren't listed in your code submission?

No

9. How did you evaluate performance of the model other than the provided metric, if at all?

I used the provided metric, as well as the AUC-ROC and balanced accuracy metrics.

10. What are some other things you tried that didn't necessarily make it into the final workflow (quick overview)?

N.A.

11. If you were to continue working on this problem for the next year, what methods or techniques might you try in order to build on your work so far? Are there other fields or features you felt would have been very helpful to have?

I would implement the Sparse-CNN aggregator, and input the information about the Breslow measure in the training process. I would also take advantage of more advanced tile-level self-supervised encoders, such as in this work in this work.

12. What simplifications could be made to run your solution faster without sacrificing significant accuracy?

One possible approach is to randomly select a smaller subset of tiles for encoding, as this is the primary computational limitation. Interestingly, the effectiveness of

this method plateaus rapidly with an increase in the number of tiles utilized. By using just 10% of the tiles per slides, we can maintain satisfactory performance, resulting in an approximate inference time of 15 minutes for 500 WSIs.

(optional) Whole Slide Images can be challenging to work with due to their size and the significant variation in their contents. What techniques and/or tools did you find helpful for working with WSIs and why?

I work mainly with small tiles and their coordinates, extracted from WSIs at a given magnification rate.