TELECOM PARIS

IMA206 PROJECT

Mixing weakly supervised and self-supervised learning techniques for melanoma relapse detection: a comparative study

Group:
Hajer Ben Ammar
Souheib Ben Mabrouk
Mayssa Haddar
Imen Mahdi

Supervisors:
Pietro Gori
Ali Mammadov

Contents

1 Introduction

This report presents a research project based on the VisioMel challenge that is aimed at improving melanoma relapse prediction. Melanoma relapse prediction is a challenging and often unreliable task, where diagnostic accuracy can vary from one healthcare professional to another. To address this challenge, more recent advances in self-supervised learning (SSL) and weakly supervised learning (WSL) have shown great promise in improving the accuracy of cancer relapse detection.

In this work, we specifically explore combinations of SSL and WSL techniques, namely BYOL (Bootstrap Your Own Latent) followed by AbMIL (Attention-based Multiple Instance Learning) and SimCLR (Simple Framework for Contrastive Learning of Representations) followed by AbMIL. BYOL and SimCLR are SSL techniques that train models using unlabeled data, while AbMIL is a WSL approach that exploits weakly labeled data without pixel-level annotations.

Through a comparative analysis, we evaluate the performance of these model combinations against traditional diagnostic methods. Our aim is to minimize the reliance on extensive human supervision and improve the reliability of the diagnostic process for melanoma relapse prediction. The results of this study demonstrate the potential of combining SSL and WSL techniques to improve predictive accuracy, addressing the limitations associated with conventional diagnostic approaches.

By leveraging the strengths of BYOL and SimCLR for self-supervised learning, as well as AbMIL for weakly supervised learning, our research provides insight into the effectiveness of using these techniques in the context of melanoma relapse prediction.

2 Problem Modeling and Data Understanding

Melanoma is a type of skin cancer that arises from melanocytes, which are cells responsible for producing pigments in the skin. It is known for its potential to metastasize meaning that cancer cells can spread from the primary tumor to other parts of the body and can be life-threatening if not detected and treated early. Furthermore, even if the initial cancer is successfully treated and removed, there is a risk of relapse and that's the main focus of our project.

To predict relapse, we need to extract crucial features from patient medical images and train a model. However, direct analysis is impractical due to the large image sizes and hardware constraints. Moreover, the varying sizes of images pose challenges for Deep Learning methods. A potential solution is to divide the images into patches, enabling workable smaller sizes. Nevertheless, the absence of patch labels and the presence of non-tumoral patches within tumor images complicate the task.

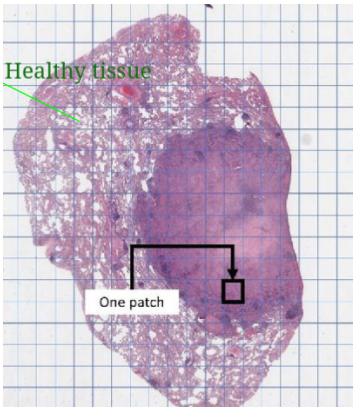


Figure 1: Patched Melanoma Image

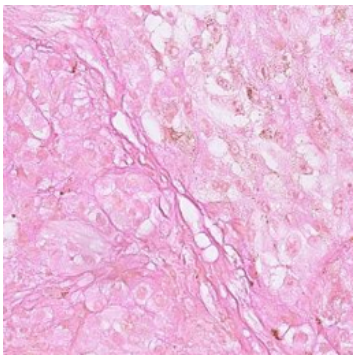


Figure 2: Random patch from a patient with melanoma relapse

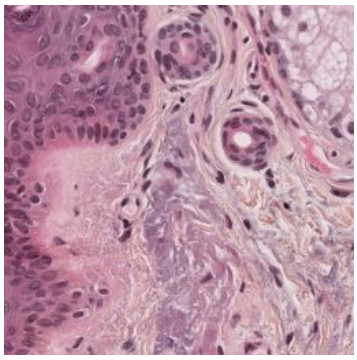


Figure 3: Random patch from a patient without melanoma relapse

While manual patch annotation is time-consuming, research has introduced a promising solution: self-supervised learning to extract features from unlabeled data and weakly-supervised learning to train the prediction model using the previously extracted features as input and the labels of the whole images. This approach eliminates the need for manual annotation and offers an efficient strategy for our task.

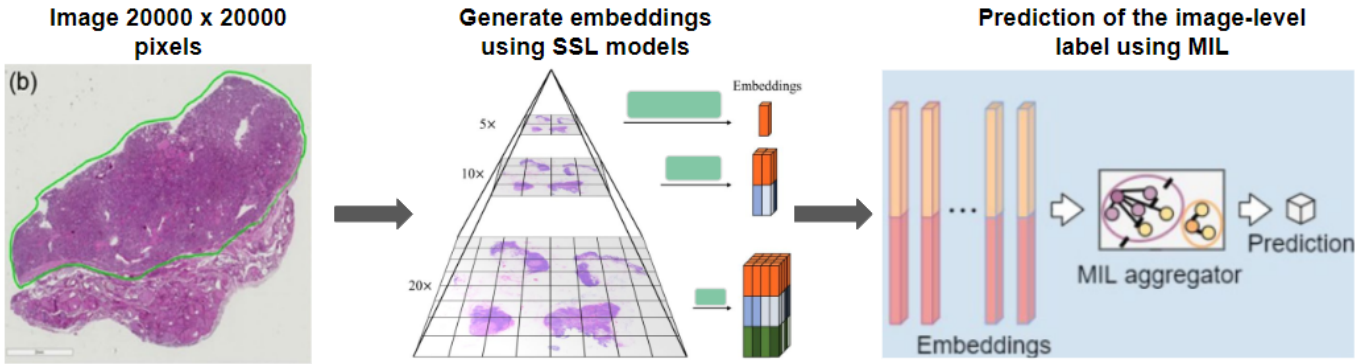


Figure 4: The main pipeline of the project

3 Feature extraction with self-supervised learning methods

Self-supervised learning (SSL) is a machine learning paradigm that enables models to learn useful representations from unlabeled data without requiring manual annotations. Unlike traditional supervised learning, where each data sample is labeled with a corresponding target, SSL uses the natural patterns and repetitions in the data to generate substitute supervision signals.

The key idea behind SSL is that by learning from a large amount of unlabeled data, models can develop a rich understanding of the underlying data distribution and capture useful representations. These learned representations can subsequently be used as feature extractors for various supervised tasks, leading to improved performance with limited labeled data.

SSL has gained significant attention in recent years due to its ability to leverage the vast amounts of unlabeled data available and its potential for addressing the data labeling bottleneck in many domains,

especially in the medical domain. It has shown promising results across various domains including computer vision.

By utilizing self-supervised learning techniques like BYOL (Bootstrap Your Own Latent) and SimCLR (Simple Framework for Contrastive Learning of Representations), we can train models to extract meaningful and informative features from unlabeled data, leading to improved performance on downstream tasks like predicting the possibility of having Melanoma relapse, which is the focus of our work.

In the context of our project, we will utilize both simCLR and BYOL techniques to explore and compare their respective performance and outcomes.

3.1 SimCLR

Simple Framework for Contrastive Learning of Representations (SimCLR) [1] is a method that utilizes various augmentations techniques to extract powerful features from unlabeled images. This technique minimizes the distance between feature vectors of two similar images and maximizes it for images that are from different classes, thus the term "Contrastive". SimCLR works by generating two views of the same image that have been augmented differently. For example, Figure 5 shows 3 different crops that have been randomly augmented. It it important to note that augmentation plays a major role in the success of such methods. Because the model is unaware of the labels of the data, we are enforcing it to not focus on color, orientation or pixel wise similarities. In fact, by changing the color, translating and rotating the crop area, and adding random noise, we are asking the model to find the features of the image that are independent from these variations. This is mimicking the way our mind recognizes objects in an image. For the example shown in Figure 5, it is possible that we recognize the dog when we look at the tail, the form of the ears, and the hanging tongue. We want our model to be able to perform similarly by generating feature vectors that capture this type of information from an image.

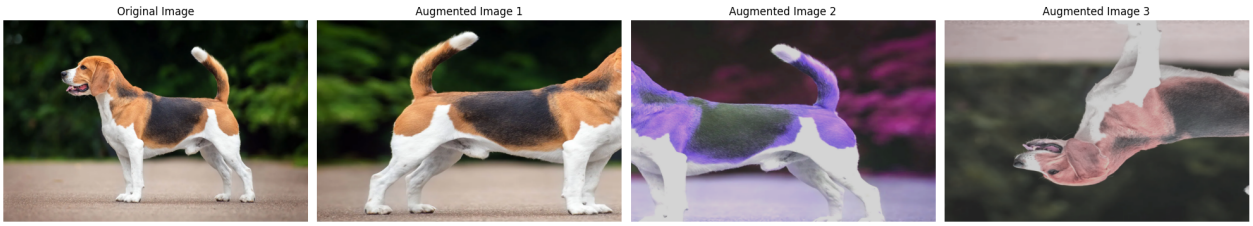


Figure 5: An example of randomly applied augmentations

SimCLR is implemented as shown in figure 6. Two randomly augmented views are fed to two networks (the choice of the architechture can vary) that will project the views into a latent space. We call these the positive samples. The same process is performed on another image to obtain the negative samples. The resulting representations will then be fed into small MLP networks that are called projection heads. Finally, the loss function will be designed such that it minimizes the distance between the output of the networks for the positive pairs and maximizes between the positive and negative pairs.

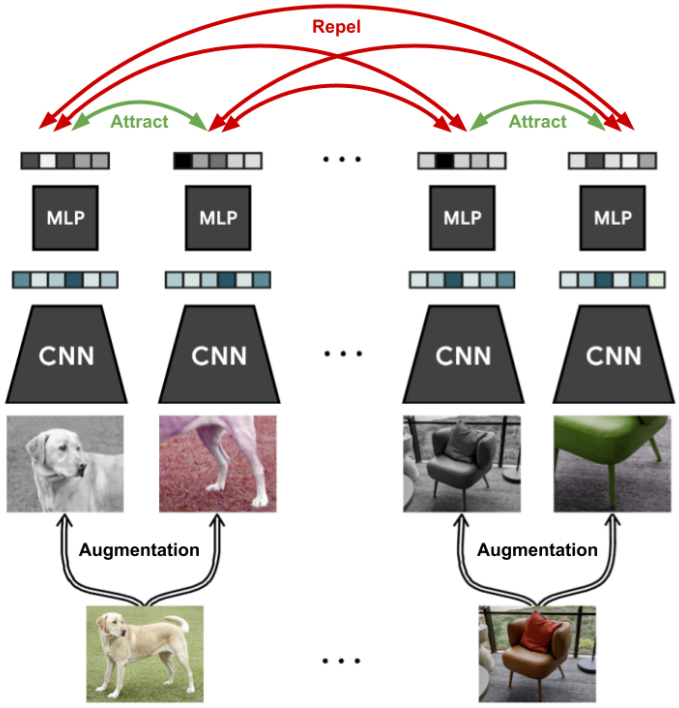


Figure 6: SimCLR Architecture

Using the negative pair is crucial for the success of the method. Otherwise, the network can collapse and output a constant vector for all images which does indeed minimize our loss function. However, the choice of the negative and positive examples in our case is challenging. Because we are using patches of the cell image, we do not know which patches belong to the healthy tissue and which do not. Therefore, for an image labeled as relapse, some healthy patches can be represented as negative. On the other hand, healthy patches from the non relapse images will be labeled as positive. The model will then repel two patches of the same nature. This could potentially lead to instability in the performance of the model.

This is why more methods that do not require negative examples were also considered.

3.2 BYOL

Unlike other contrastive learning methods like simCLR, **B**ootstrap **Y**our **O**wn **L**atent (BYOL)[2] achieves state-of-the-art performance without using any negative samples. Fundamentally, BYOL uses two neural networks referred to as **online and target networks**, which share the same architecture but do not have the same weights and learn from each other. By presenting an augmented view of an image, the online network is trained to predict the target network's representation of the same image under a different augmentation. By doing so, we can train a new and potentially improved representation known as the online. Simultaneously, the target network is updated using a slow-moving average of the online network, which helps stabilize the learning process and allows the target network to provide a more consistent target for the online network's predictions. This iterative process enables BYOL to bootstrap and enhance image representations through interaction and prediction. These learned feature representations capture meaningful and discriminative information about the images, which can be used for prediction.

Figure 7 shows a representation of BYOL's architecture.

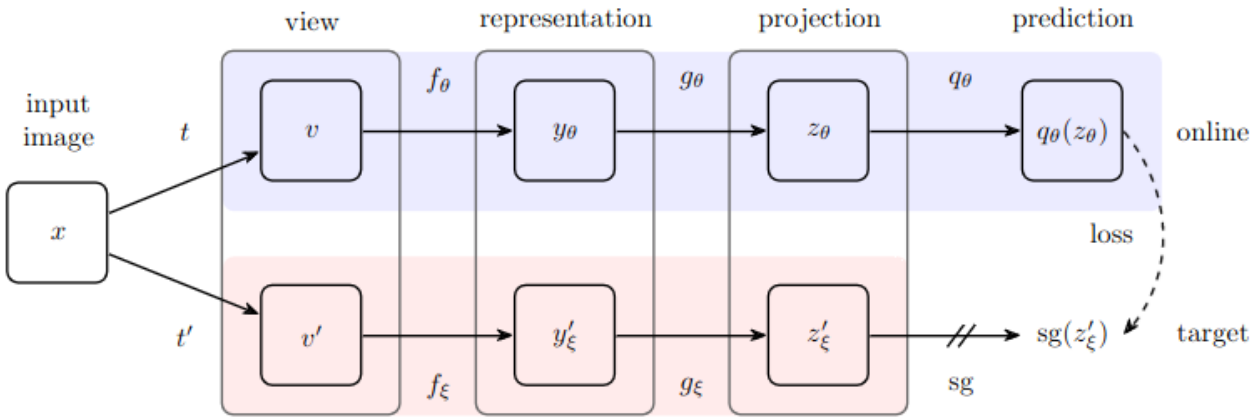


Figure 7: BYOL's architecture.

BYOL's goal is to learn the representation y_θ. After each training step, the weights of the target network are updated as follows: $\xi \leftarrow \tau\xi + (1-\tau)\theta$.

Each of the two networks (online and target networks) outputs a representation and a projection from one of the two differently augmented views of the image which were produced by BYOL. After that, we output the prediction $q_\theta(z_\theta)$ of z'_ξ and ℓ_2-normalize both of them. And finally, we define the mean squared error between the normalized predictions and the target projections:

$$\mathcal{L}_{\theta,\xi} \triangleq \left\|\overline{q_\theta}(z_\theta) - \overline{z}'_\xi\right\|_2^2 = 2 - 2 \cdot \frac{\langle q_\theta(z_\theta), z'_\xi \rangle}{\left\|q_\theta(z_\theta)\right\|_2 \cdot \left\|z'_\xi\right\|_2}.$$

The output of BYOL, which represents learned features, will be further utilized in the next step of the project as input to the prediction model, enhancing its performance by leveraging the enriched representations obtained through self-supervised learning.

4 Weakly supervised learning with Multiple-Instance Learning

Once the features are extracted using the self-supervised models, we use weakly supervised learning methods, like Multiple Instance Learning, to predict the possibility of having a Melanoma relapse.

Weakly supervised learning, and more specifically Multiple Instance Learning, are considered a valuable approach in multiple medical imaging tasks. Weakly Supervised Learning deals with a combination of a small amount of labeled data and a large amount of unlabeled data which fits the description of the data we are dealing with. In fact, having only one label per whole slide image, all of the patches that make up the image are unlabeled.

Multiple Instance Learning, also known as MIL, is a form of weakly supervised learning, for which training instances are organized into sets, referred to as bags. Unlike traditional supervised learning, where labels are assigned to individual instances, MIL assigns a single label to each bag. In the context of predicting Melanoma relapse, the whole slide image serves as the bag, while the patches comprising the image represent the instances.

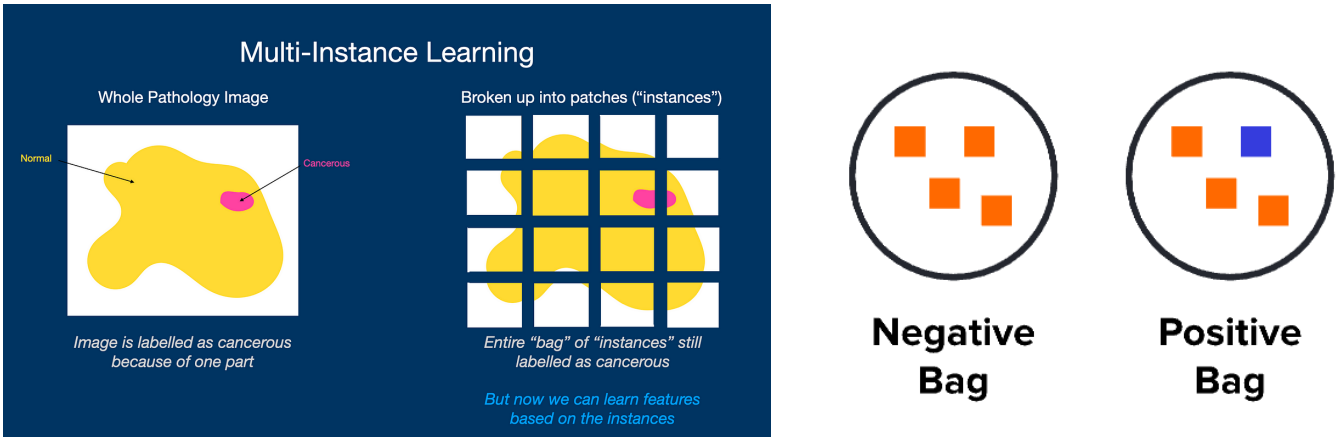


Figure 8: MIL using the slide as a bag of patches

4.1 State of the art methods

State-of-the-art methods in the field of MIL encompass various models, including dual-stream MIL, CLAM and attention-based MIL. These models have demonstrated great potential while dealing with features extracted with self-supervised models in addressing weakly supervised learning tasks.

4.1.1 Dual-Stream MIL

One prominent method is the Dual-stream Multiple Instance Learning method[4]. This model features two parallel streams of information processing.
The first stream is focused on global information at the bag level, while the second places the emphasis on local information from individual instances within the bags. By taking into account both global and local context, Dual-Stream MIL improves the model's ability to capture macroscopic and microscopic features relevant to the task in hand.

4.1.2 CLAM : Clustering-constrained Attention MIL

CLAM [5] is a deep learning technique, more specifically a weakly supervised learning technique, that leverages attention-based learning. It automatically detects specific sub-regions within an input, such as a whole slide, that hold significant diagnostic information.

Furthermore, CLAM incorporates instance-level clustering to limit and improve the feature space, using the representative regions it identified earlier.

Using the whole slide image ground truth and the attention scores predicted by its attention mechanism, it generates pseudo-labels for both highly frequented and sparsely frequented patches, thus creating a new means of augmenting supervision.

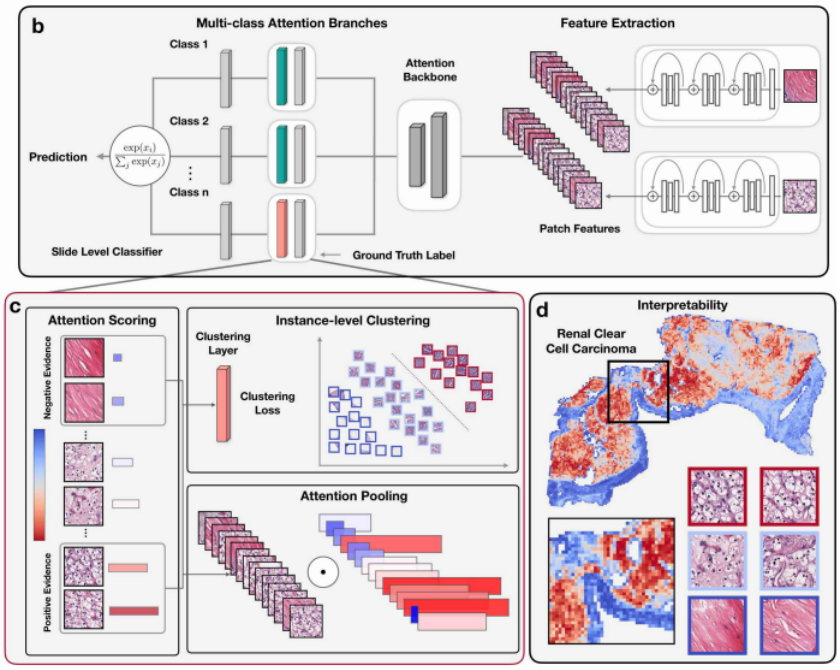


Figure 9: CLAM's architecture.

4.2 Attention-based Deep MIL

Another state-of-the-art method is the Attention-based Deep Multiple Instance Leaning [3]. This model leverages attention mechanisms in order to focus on the most informative instances within the bags. This is done by assigning high weights to patches that are most relevant, in other words, patches that contribute the most to the decision-making on the whole bag. This not only enables improved prediction accuracy compared to other methods, but it also offers better interpretability than other MIL methods. In fact, using MIL, we assign one prediction on each slide image. Knowing the different weights generated by attention-based MIL, we are able to check which patches contributed the most in the decision. This point is important, especially for medical-related tasks.

For the purpose of our project, we will be using this method thanks to the explainability of the predictions. In this method, the problem is expressed as learning the Bernoulli distribution of the bag label. The goal is to estimate the probability of the bag label. Neural networks are used in order to parameterize the distribution. We assume that individual labels exist for the instances within a bag, i.e., $y_1, ..., y_K$ and $y_k \in \{0, 1\}$, for $k = 1, ..., K$, and that there is no access to those labels. We can re-write the assumptions of the MIL problem in the following form:

$$Y = \begin{cases} 0, & \text{iff } \sum_k y_k = 0 \\ 1, & \text{otherwise} \end{cases} \tag{1}$$

For our work, we used the gated attention model as it is designed to selectively attend to informative instances within a bag while suppressing irrelevant or noisy instances during the prediction process. The gated attention model is more reliable when learning complex relations between instances than the simple attention model.

The GatedAttention model consists of two main components: a gating network and an attention network.

1. **Gating Network:** The gating network is a feed-forward neural network that takes as input the features of the instances within a bag and outputs the gating coefficients of each instance. These gating coefficients showcase the discriminative properties of the instances.

2. **Attention Network:** The attention network is a feed-forward neural network that takes as input the features of the instances within the bag and the gating coefficients and outputs the attention weights for each instance based. The more relevant an instance is deemed in the final decision, the higher the weight it is assigned.

The multiplication of both coefficients gives us the final weights we will be working with.

5 Implementation

5.1 Self-supervised learning methods

5.1.1 Solo-learn library

Solo-learn is an open source library of self-supervised methods for unsupervised visual representation learning powered by PyTorch Lightning that offers a wide variety of state-of-the-art methods. It is easy to implement and flexible in the choice of network architecture and training parameters. It is also easily connected to Weights and Biases which offers visualization, monitoring and interpretation of the results. With solo-learn, we were able to easily implement and compare different architectures and parameter combinations.

5.1.2 Implementation from scratch

To further our understanding of these methods how they work, we decided to also include our own implementations of the models. Our main references were the pseudo-codes provided by the original papers. We also looked into some open source implementations to further improve our code. The loss function implementation was also inspired by code provided in [3].

Additionally, we investigated different variations for the transformations applied in augmentation and evaluate their effect on our specific dataset which we will discuss in the results section 6.

5.2 Weakly-supervised learning with AbMIL

The feature extraction algorithm based on the ResNet-18 architecture was the initial step in our data processing pipeline. It allowed us to extract representative features from image patches using a pre-trained ResNet-18 model as the feature extraction backbone. We built a model by combining the backbone with a flattening layer, converting the output feature maps into a compact 1D feature vector.

In a subsequent step, we improved the feature extraction process by incorporating the previously discussed SSL models 5.1.1, enabling us to exploit self-supervised learning techniques to extract more robust and discriminative features from image data [1]. The combination of the initial ResNet-18-based algorithm and SSL models provided a comprehensive and efficient feature extraction framework for our dataset. As a result, the embeddings were stored in CSV files, for the training of the MIL model.

Algorithm 1 Feature Extractor

Input: $N : number\ of\ slides$; $Slide\ S_i\ contains\ n_i\ patches$; $model_{SSL}$ =pretrained SSL model
Output: *Embeddings* (Features extracted from slides in csv file) =0

for $k \leftarrow 1$ to N **do**
$embeddingList = [\,]$
  for $j \leftarrow 1$ to n_k **do**
$p \leftarrow$ patch j of the slide k
$p \leftarrow$ preprocess(p)
$embedding \leftarrow model_{SSL}(p)$
$embeddingList$.append($embedding$)
  end for
save($embeddingList$) as CSV file
end for

The implementation of Gated Attention consists of two linear layers followed by activation functions to calculate attention weights. These weights are then used to average the instances in each bag, producing a bag-level representation. The classifier is applied to the bag-level representation to predict the slide label. In this section, we define the training loop, in which the MIL network is trained using Adam's optimizer and a binary cross-entropy loss with class weights.[2]

At each epoch of the training loop, the data are randomly shuffled to induce randomness in the training process. We notice that the use of the dropout technique improves training and reduces runtime by around 5%.

Algorithm 2 MIL training

Input: N : number of Embeddings; K : number of epochs, *GatedAttention* : class for the model
Output: $model_{MIL}$ (trained MIL model) =0

function TRAIN(model$_{MIL}$, N)
  $data_{train}$ ← Load the training embeddings
  shuffle(data$_{train}$)
  Droupout some patch-embeddings from each slide
  criterion ← BCE weighted Loss
  $loss$ ← 0.0
  for slide in data$_{train}$ **do** :
    $loss$ ← $loss$ + CalculateObjective(model$_{MIL}$, $slide$, $label$)
  end for
  Backpropagation(model$_{MIL}$)
return(model$_{MIL}$)
end function

model$_{MIL}$← GatedAttention()
for k ← 1 to K **do**
  model$_{MIL}$← TRAIN(model$_{MIL}$, N)
end for

6 Results Analysis and Discussion

In this section, we will be discussing the results obtained during the different levels of our work.

6.1 Augmentation

As mentioned in previous sections 3.1 and 5.1.2, augmentation plays a critical role in the success of the SSL methods. We can control what we want the model to focus on by modifying the transformations applied to the images accordingly. Therefore, part of our project was focused on visualizing the augmented images and analyzing their possible impact on the performance for a more efficient implementation.

In the case of Figure 5, the cropping it is possible to represent only one part of the object (dog face, legs, or paws). This is because we can still distinguish a dog in an image by looking at one part of it. We then simply resize the crop to the original shape because the size does not matter in this case. However, for cell patches, this is more challenging. The dimensions of the cells can be an important indicator for the diagnosis. We can not remove the cropping from the transformations as well because we want our model to not focus on pixel-wise similarities. Therefore, the cropping had to be performed but kept limited to no more than 90% of the original size.

Color can also be an important indicator for differentiating between cells. We therefore limited the color transformations applied. Additionally, We added random Gaussian noise and rotations for more robust representations. An example of the final augmentations can be shown in Figure 10.

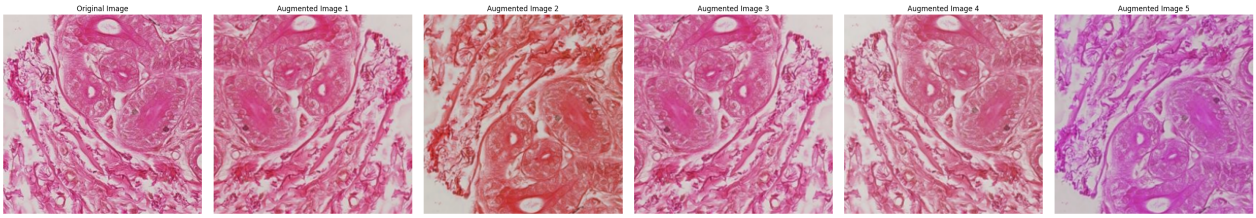
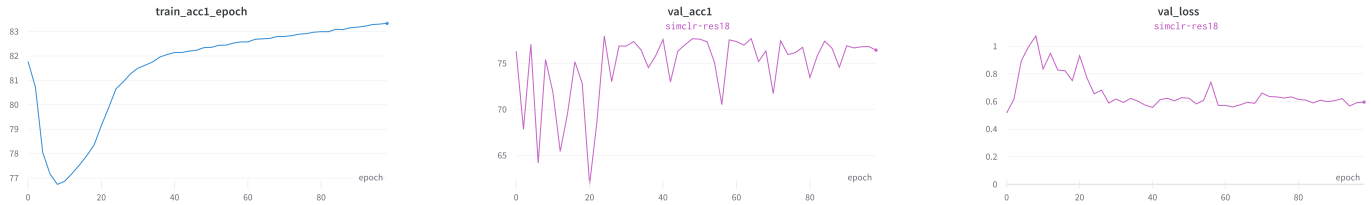


Figure 10: An example of randomly applied augmentations to a cell patch

6.2 SSL

Using solo-learn implementations, we trained SimCLR and BYOL models with Reset18 as backbones for 100 epochs. For SimCLR, the training lasted 3d 19h 39m 9s on a GPU.
For SimCLR, we used the pretrained weights as a starting point. The results are shown in Figure 11. We can see that for the first 10 epochs, performance drops significantly and then improves progressively over the following epochs. However, we notice that after 100 epochs the validation accuracy does not improve over the initial one. We do notice that the train accuracy continues to rise indicating possible over-fitting if we continue the training.

It was interesting to see that fine-tuning the model on our images did not outperform the initial pretrained model. This can be due to the ambiguity when defining the negative and positive pairs in the model. Therefore, the model is possibly improving only because it collapses into a constant feature vector.



((a)) Training accuracy ((b)) Validation accuracy ((c)) Validation loss

Figure 11: Results of training a SimCLR with resnet18 over 100 epochs

6.3 MIL

We encountered numerous challenges during the training process of the MIL using features extracted from various models. After careful analysis and deliberation, we have determined that one of the causes of these problems is the highly unbalanced nature of the data we are dealing with (83% of the data corresponds to the label 0). In order to address this issue effectively, we have devised distinct AbMIL algorithms that yield different outcomes. Specifically, one algorithm involves weighting the loss function using the proper proportions of each class, and shuffling the data, while the other incorporates some additional dropout techniques.
Another issue we faced was the long running time of the algorithm. Using the dropout techniques AbMIL helped us reduce the running time by approximately 5% (1 hour over 23).

After the training of the attention-based MIL on features extracted by Resnet18 and by BYOL, we conducted a comparative study to examine the obtained results. For this purpose, we visualized both the training and validation loss on all epochs along with the validation accuracy.

- **AbMIL trained on features extracted with Resnet18 vs Byol**

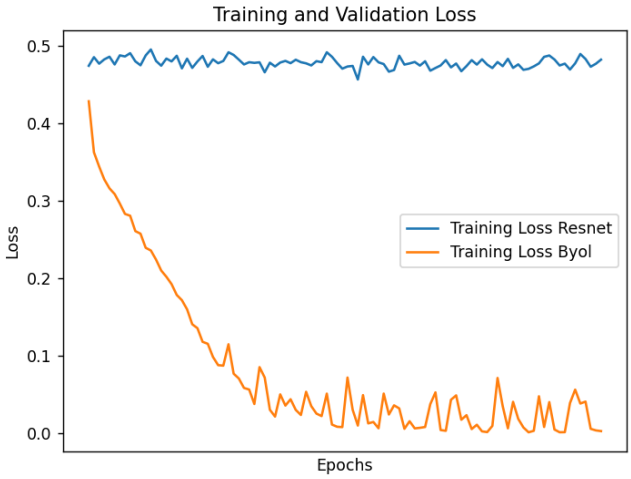


Figure 12: Training Loss Byol VS Resnet

We can notice that we do not have good results with Resnet18. In fact, the AbMIL model was not able to differentiate the features vectors of negative and positive individuals that's why the training loss was constant all over the epochs. Otherwise, the feature quality was improved by the usage of BYOL. The model was able to train and the training loss decreases from one epoch to another.

- **AbMIL trained on features extracted with Resnet18**

  From the graphs below, we can clearly see that the obtained results are less than satisfactory with Resnet18. In fact, while applying the AbMIL with the shuffled data, we obtain a constant accuracy value of 0.83. While looking at the value itself, one might say that it's not a bad result, however, the fact that the accuracy value never changes between epochs is an indication of overfitting on the dominant class. In the case of AbMIL with dropouts, there was a slight fluctuation in the accuracy but it soon goes back to being constant. The training loss value in both cases is significantly high, between 0.45 and 0.5, and the validation loss value reaches even higher values, in the order of 0.8 in some epochs.

  We can explain these results by a number of different hypothesizes:

  – While inspecting the data values generated by Resnet18, we found that the row and column values of the CSV files are too close, generally varying from 0.7 to 1.4 in small intervals. In other words, different image patches were embedded into similar feature vectors. Therefore, Resnet18 may not be an excellent backbone to use in this case study and we need larger feature vectors. The similarities of the extracted features may lead our MIL model to have difficulty differentiating between the two classes.

  – Another possible explanation is that two layers of AbMIL are not enough to capture significant information from the complex data.

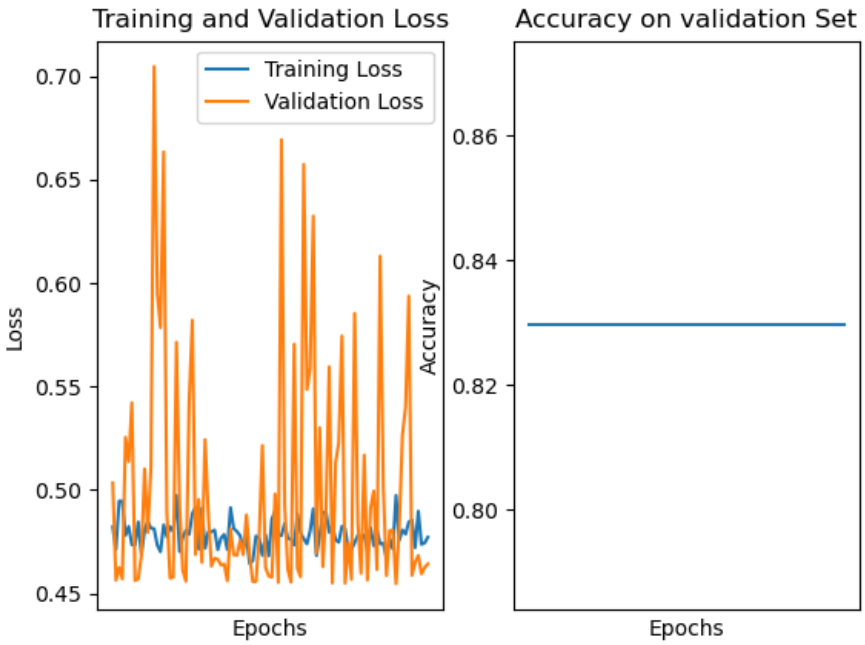


Figure 13: Results obtained with the combination of Resnet18 and AbMIL with shuffled data

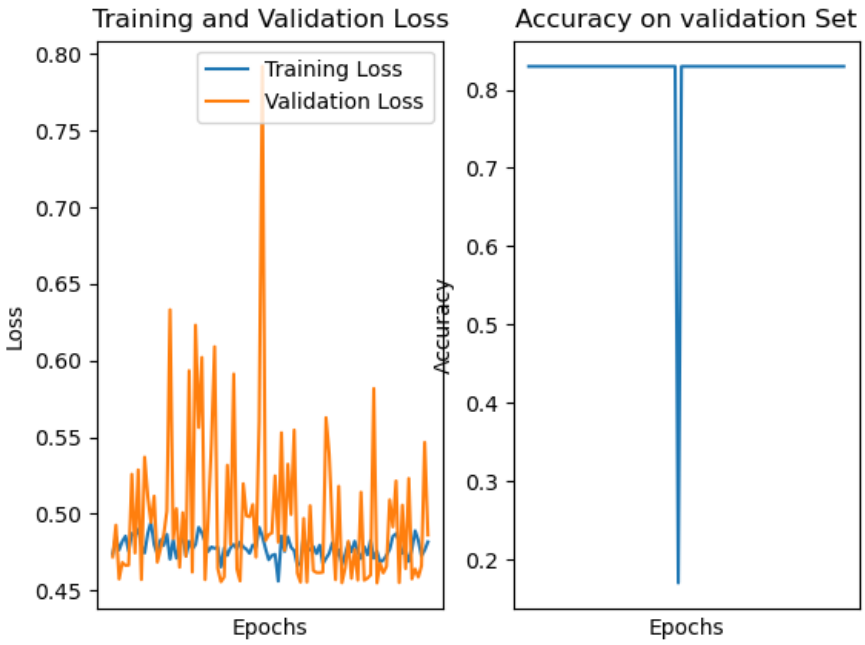


Figure 14: Results obtained with the combination of Resnet18 and AbMIL with dropouts

- **MIL trained on features extracted with BYOL**

  Based on the provided graphs, it is evident that there is a slight improvement in the results compared to the previous outcomes achieved with ResNet18. However, they still fall short in terms of meeting the desired outcome.

Notably, the training loss exhibits a significant reduction, with a value close to zero. On the other hand, the validation loss is notably worse compared to the previous results, particularly in the case of AbMIL with dropouts, where it reaches a value of 2. The validation accuracy remains consistent with the previous outcome, with a constant accuracy value of 0.83. This indicates that the model consistently favors the majority class, rather than achieving more balanced predictions.

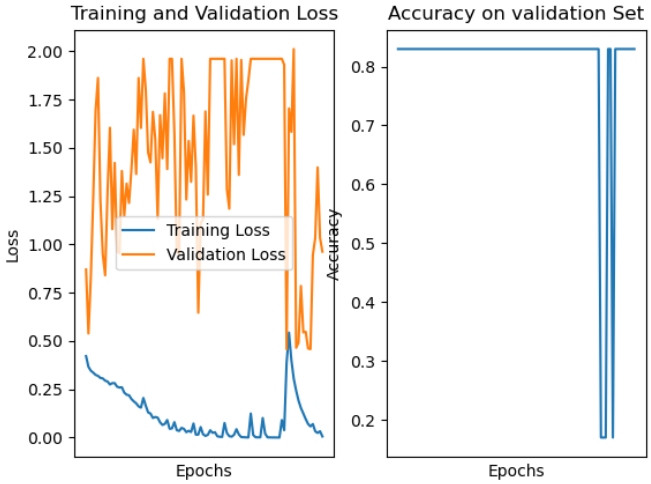


Figure 15: Results obtained with the combination of BYOL and AbMIL with shuffled data

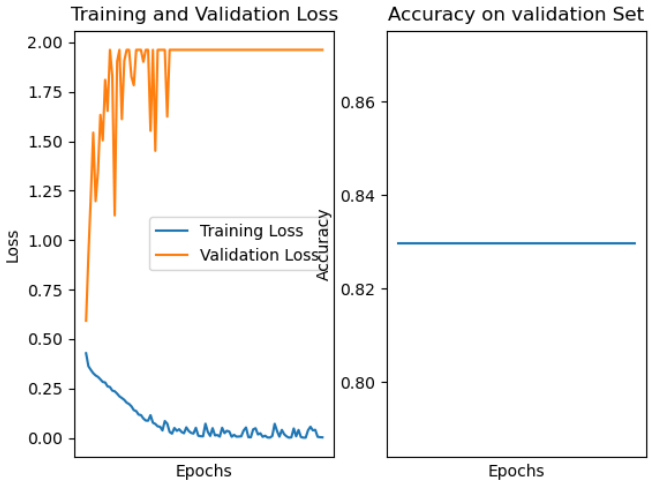


Figure 16: Results obtained with the combination of BYOL and AbMIL with dropouts

7 Limitations and Potential Improvements

Applying SSL methods and evaluating them using MIL techniques proved very challenging for our specific dataset. First, SimCLR uses negative pairs to avoid collapses in the network outputs which in our case caused some ambiguity in the way we define the negative and positive pairs. To solve this, we used BYOL which does not require contrastive learning and thus does not need defining the negative pairs. Transformers are also a good method that allows for more interpretability of the results.However,

training SSL models was intensive in terms of computations and training duration which made it difficult to try different combinations and compare their results.

After evaluating the performance of the MIL model on feature vectors extracted using BYOL, we did notice an improvement over simply using a pretrained ResNet18. However, we could not achieve satisfactory results. This could be because the chosen architectures were not complex enough for the task at hand.

8 Conclusion

In conclusion, our project findings emphasize the significance of augmentation in semi-supervised learning (SSL) and highlight the importance of tailoring augmentation techniques to match the specific task and dataset being utilized.

Regarding the dataset used, we found that contrastive learning is not suitable due to the inherent ambiguity in defining negative pairs, leading to suboptimal results. Therefore, alternative approaches should be explored for this particular dataset.

Furthermore, our work indicates that melanoma detection is a complex task that presents significant challenges when attempting to solve it using simple architectures like ResNet18. To achieve improved performance, it is crucial to consider more sophisticated and advanced architectures or explore additional methods specifically designed for tackling the complexities of melanoma detection including transformers and and extending training durations.

References

[1] Ting Chen et al. "A Simple Framework for Contrastive Learning of Visual Representations". In: *ArXiv* abs/2002.05709 (2020).

[2] Jean-Bastien Grill et al. "Bootstrap your own latent-a new approach to self-supervised learning". In: *Advances in neural information processing systems* 33 (2020), pp. 21271–21284. URL: https://arxiv.org/pdf/2006.07733.pdf.

[3] Maximilian Ilse, Jakub M. Tomczak, and Max Welling. *Attention-based deep multiple instance learning*. June 2018. URL: https://arxiv.org/abs/1802.04712.

[4] Bin Li, Yin Li, and Kevin W. Eliceiri. *Dual-stream multiple instance learning network for whole slide image classification with self-supervised contrastive learning*. Apr. 2021. URL: https://arxiv.org/abs/2011.08939.

[5] Ming Y. Lu et al. *Data efficient and weakly supervised computational pathology on whole slide images*. May 2020. URL: https://arxiv.org/abs/2004.09666.