Scaling mega-pixel images classification via attention sampling

Lymphoma subtype case study

Alberto Sinigaglia
July 3rd 2023

Introduction

- Lymphoma is a type of cancer that accounts for 4% of cancer diagnoses, having affected over half a million people
- Lymphoma subtype classification strongly determines the patient treatment
- Three main types of lymphoma: Chronic Lymphocytic Leukemia (CLL), Follicular Lymphoma (FL), and Mantle Cell Lymphoma (MCL)
- The diagnosis is obtained by experts analysing high resolution images from exams of the patient, requiring high level of expertise

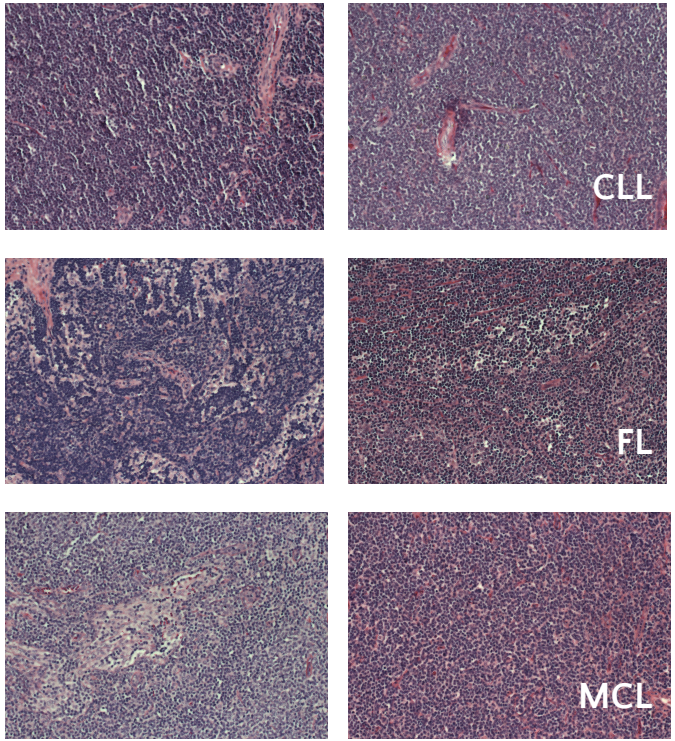
Reasons

- Governments are starting large scale screening campaigns for the prevention of such diseases
- The human-in-the-loop part of the screening is highly expensive and error prone
- In order to allow large scale screening in a wide variety of diseases, automatic procedures are necessary
- Current proposed solutions merely address the accuracy side of the problem, do not consider the deployment and explainability

Dataset

374 high resolution images 1388 x 1040:

- 113 for the Chronic Lymphocytic Leukemia (CLL) class
- 139 for the Follicular Lymphoma (FL) class
- 122 for the Mantle Cell Lymphoma (MCL) class



SoTA

Current proposed methods can be subdivided in 2 categories, ML and DL based.

Machine Learning based relies on experts' opinions on relevant features, and automated techniques for use of such features

Pro: fast and expert based
Cons: suboptimal, not explainable out of the box

Deep Learning based relies on neural networks for an automatic feature extractor and classification

Pro: potentially optimal and more powerful
Cons: computationally heavy, not explainable

We will consider only the latter one, trying to solve the listed issues .

DL SoTA

The methods proposed in the literature mainly differ in the architecture as source of improvement, trying to induce some bias to drive the network to have a better performance.

Usually based on the following steps:

1. Given an image **I**
2. Sample **N** subpatches (for computational reasons)
3. Forward them through the Neural Network
4. Use some decision fusion techniques to derive the final classification

Issues

However, many issues arise from this way of reasoning:

1. Uniform sampling of the patches induces a uniform prior over the distribution of useful information in the image
2. Proposed models are becoming too computational costly due to their size
3. All of the decision fusion techniques proposed (most frequent prediction, Bayesian averaging, weighted voting) assume equal importance from the patches
4. No attention is being given to the explainability of the problem, thus relying strictly on the classification

Breaking the prior (1 / 2)

Introducing a patch sampling step changes the MLE procedure:

$$\max_{\theta} \prod p_\theta(y_i|x_i) \Rightarrow \max_{\theta} \prod \mathbb{E}_{s_i \sim x_i}[p_\theta(y_i|s_i)]$$

Assuming a uniform prior for the patches, we can rewrite it to:

$$\max_{\theta} \prod_{(x_i,y_i)} \mathbb{E}_{s_i \sim x_i}[p_\theta(y_i|s_i)] = \max_{\theta} \prod_{(s_i,y_i)} p_\theta(y_i|s_i)$$

Thus converting it to log-likelihood, it leads to the usual optimization objective:

$$\max_{\theta} \prod_{(s_i,y_i)} p_\theta(y_i|s_i) \propto \min_{\theta} \sum_{(s_i \sim x_i,y_i)} -\log p_\theta(y_i|s_i)$$

Breaking the prior (2 / 2)

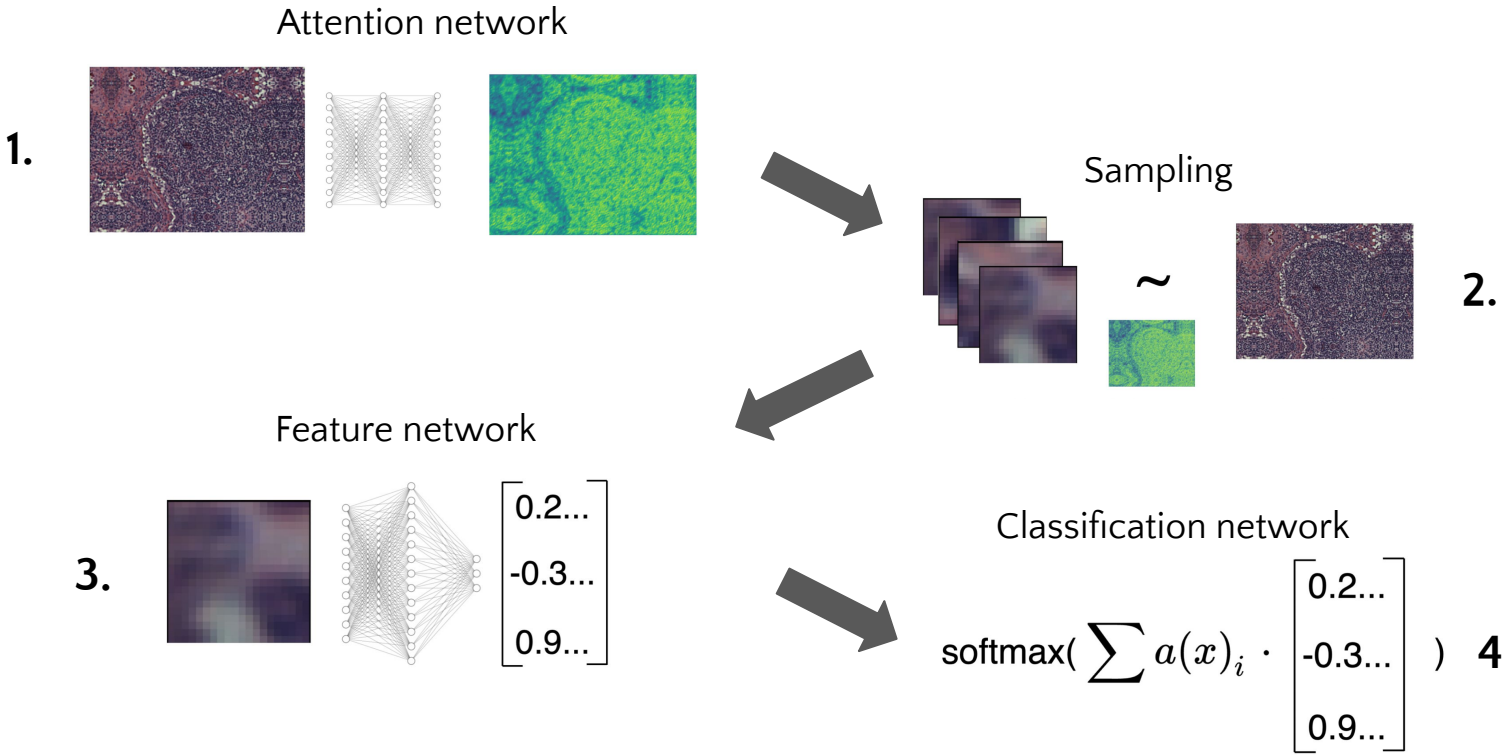
However, estimating the posterior for sampling, can lead to much more useful patches

$$\max_\theta \prod p_\theta(y_i|x_i) \Rightarrow \max_\theta \prod \mathbb{E}_{s_i \sim x_i}[p_\theta(y_i|s_i)]$$

$$\approx \max_\theta \prod_{(x_i,y_i)} p_\theta(y_i|s_i)p(s_i|x_i)$$

In order to do so, one Neural Network will learn the classification likelihood, and a second one will try to learn the patches' posterior.

4 stage architecture



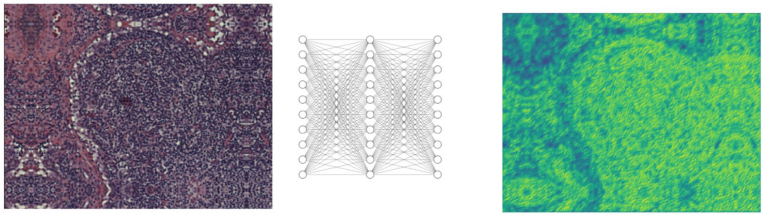
Attention Network

Responsible for learning a useful posterior, thus allowing the classifier to be fed only useful patches of the image, not assuming anything about the distribution.

Inspired by PatchGAN, it takes as input a downscaled version of the image, and then maps it to a distribution using 3 blocks composed by:

- Convolution 3x3 (8 filters)
- BatchNorm

And a final 3x3 Convolution with 1 filter and bi-axis softmax activation to ensure the sum is equal to 1.

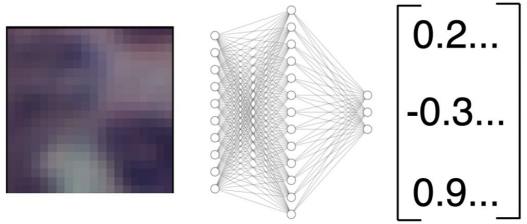


Feature Network

Responsible for learning the actual logits for the classification. However, in contrast to the usual methods, those logits are normalized at the end, thus having magnitude 1

Takes in input a patch of the image according to the posterior, and then maps it to the output with the following architecture:

- x4 Convolution 3x3 (16 filters)
- Global max pooling
- Linear fully connected layer
- Normalizing layer



Classification Network

Responsible for learning the actual classification. It can be a neural network, however to keep the model as light as possible, it's been implemented as follows:

1. takes the **P** sampled patches of the image
2. does a weighted sum of the normalized logits of each of them
3. applies a softmax to have an actual classification

This way, an importance proportional to the attention of its original patch is given to each logit vector

$$o_j = g\left(\frac{\sum_{i=0}^{P} a(x;\theta)_{\text{patch}_i} \cdot f(\text{patch}_i;\phi)}{\sum_{i=0}^{P} a(x;\theta)_{\text{patch}_i}}\right)$$

Learning the posterior (1 / 2)

The gradient cannot flow through the sampling procedure in order to estimate θ_a of the attention network, and since the sampling is done over the input, techniques such as straight-through-estimator or the reparameterization-trick cannot be applied.

$$\nabla_{\theta_a} L = \frac{\delta L}{\delta o} \frac{\delta o}{\delta f} \frac{\delta f}{\delta a} \frac{\delta a}{\delta \theta_a}$$

In order to do so, we relied on Reinforcement Learning, in particular to the Policy Gradient Theorem:

$$\begin{aligned} \frac{\delta}{\delta \theta} \sum f(x;\phi) &\approx \mathbb{E}_{p \sim a(x;\theta)} \left[\frac{\frac{\delta}{\delta \theta} a(x;\theta) f(p;\phi)}{a(x;\theta)} \right] \\ &= \mathbb{E}_{p \sim a(x;\theta)} \left[\frac{\delta}{\delta \theta} \log a(x;\theta) f(p;\phi) \right] \end{aligned}$$

Learning the posterior (2 / 2)

Using the chain rule, we can see that we just need to optimize the following objective:

$$\frac{\delta L}{\delta \theta} \approx \mathbb{E}_{p \sim a(x;\theta)}[\frac{\delta}{\delta \theta} \log a(x;\theta) L(g(f(p;\phi));y)]$$

In order to overcome the gradient variance, we will use a baseline:

$$\mathbb{E}_{p \sim a(x;\theta)}[\frac{\delta}{\delta \theta} \log a(x;\theta)(r - avg(r))]$$

In order to overcome the exploration/exploitation dilemma, we will introduce an entropy regularizer:

$$L'_a(x;y) = L_a(x;y) + \lambda \mathcal{H}(a(x))$$

$$\mathcal{H}(p) = \sum_x p(x) \log(p(x))$$

Training (1 / 2)

The training can be seen as a 2 player cooperative game, which induces a non-stationarity of the training, as the distribution of the second player, feature and classification network, will change due to its dependence on the first player, attention network.
However, independent learning seem to be working just fine for this specific problem, thus each agent will optimize its own reward function.

Assuming the marginal $p_\theta(y_i|s_i)$ is a categorical distribution, optimizing its MLE implies minimizing Cat-Cross-Entropy:

$$\min_\theta - \sum_i^{\text{classes}} y_i log(\hat{y}_i)$$

Given REINFORCE with baseline and entropy regularization, the attention network will perform GD with the following gradient :

$$\nabla_\theta \mathcal{L} = [L(\phi(s), y) - \bar{L}] \nabla a_\theta(s|x) + \lambda \sum_i a_\theta(s_i|x) \log a_\theta(s_i|x)$$

Training (2 / 2)

The architecture needs some hyperparameters:

- Entropy coefficient, as a trade-off between exploration and exploitation
- Number of patches to sample from each image
- Size of the patches

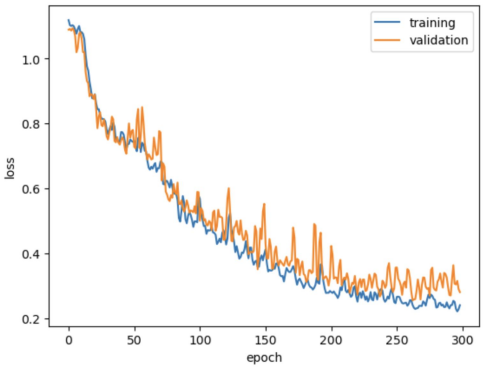
Multiple instances of the network are being trained, each of them fixing the hyperparameters and following the same procedure:

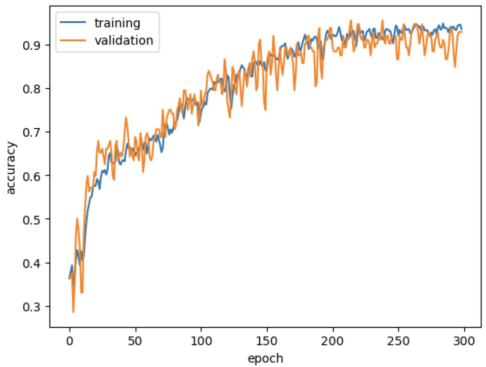
- sample batch of data from training set
- apply data augmentation
- forward pass of the network
- calculate loss
- 1 GD step using Adam with 10^{-3} step-size for feature network
- 1 GD step using Adam with 10^{-3} step-size and 0.999 as first moment decay for attention network

Results

P. size	N. of P.	Training acc.	Test acc.	Time
50	1	82% ± 2%	81% ± 2%	0:55h
50	5	90% ± 1%	89% ± 2%	1:45h
50	20	95% ± 1%	94% ± 1%	2:50h
20	5	88% ± 2%	87% ± 2%	1:25h

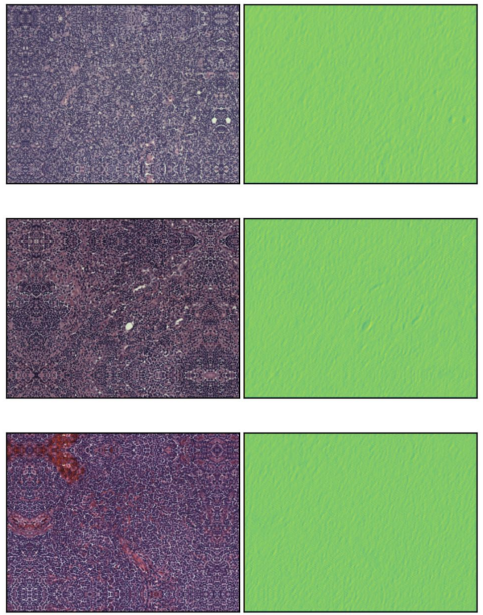
Each model has been trained 5 times with 5 different seeds to approximate a 5-fold-CV



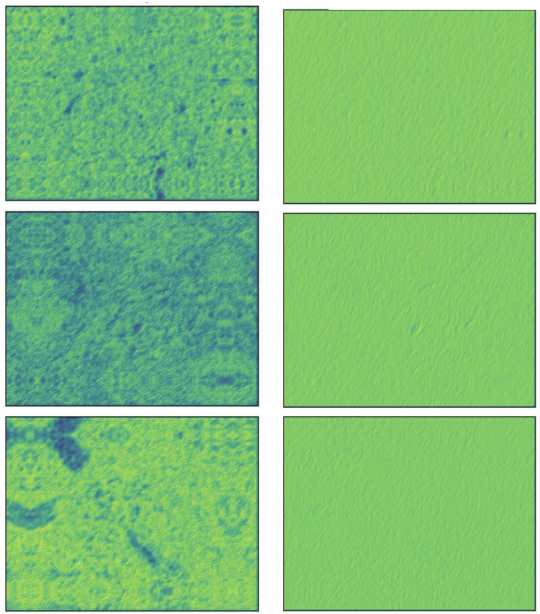


Training history of the model with 20 patches of size 50x50px

Analysis



Attention learned by the 50x50px model



50x50px 20x20px attention difference

Conclusions

The models presented in the literature are based on uniform sampling of 100x100px patches. The attentions learned by the attention network clearly explain why that method is effective in this study case, however this does not hold for localized problems, such as tumor detection.

The proposed method instead, breaks this prior belief by learning a posterior, thus learning how to sample from the original images, allowing the classifier to see only meaningful images.

Furthermore, once deployed, an expert can check if the learned attention is meaningful, thus giving more credibility to the whole architecture, improving its explainability.

Finally, once learned, the attention, as almost independent from the classifier, can be used as part of the training of a much more powerful classifier, depending on the available budget and hardware.

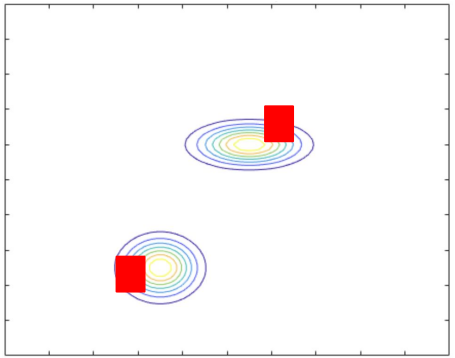
Future improvements (1 / 2)

The attention network defines a distribution over which we sample patches, thus inducing an indirect independence between patches.

$$p(y|s_1, s_2, ..., s_n)p(s_1, s_2, ..., s_n|x) = p(y|s_1, s_2, ..., s_n)p(s_1|x)p(s_2|x)...p(s_n|x)$$

Even though this might not cause problems in expectations, it might lead to a very high variance due to the restriction on the size of the patches.

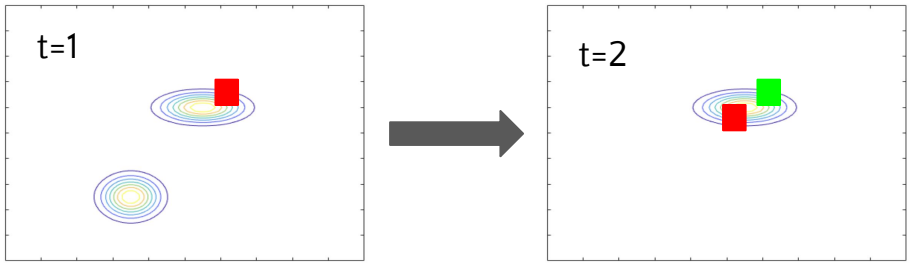
For example, in this picture there is a 50% chance that the 2 patches sampled come from different distributions, however they might not be big enough to be meaningful by themselves



Future improvements (2 / 2)

Instead of considering the patches conditionally independent, a different posterior can be learned:

$$p(y|s_1, s_2, ..., s_n)p(s_1, s_2, ..., s_n|x) = p(y|s_1, s_2, ..., s_n)$$
$$p(s_1|x)\cdot$$
$$p(s_2|x, s_1)\cdot$$
$$p(s_3|x, s_1, s_2)\cdot$$
$$... \cdot p(s_n|x, s_1, s_2, ...)$$



To fully exploit this property, it will necessitate a sequence modelling classifier with positional embedding, such as a ViT

Thanks