Lymphoma Subtype Classification
a performance-efficency trade-off approach

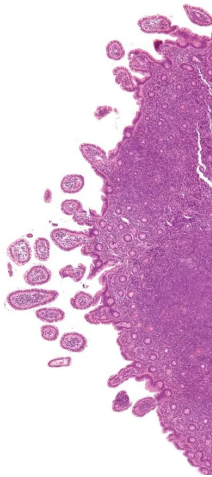
Nicola Bee Lorenzo Saccaro

22 February 2023

Università
degli Studi
di Padova

Introduction

- Lymphoma is a type of **cancer** that originates from the cells of the immune system
- It is estimated that is responsible for half of the malignant blood diseases worldwide
- In recent years **Deep Learning** techniques have shown encouraging results in the classification of histological images



Goals

Our study aims to:

- Deepen the lymphoma **classification** with AI techniques, in particular using a combination of different Convolutional Neural Networks and classical machine-learning classifiers.
- Evaluate if the **colorspace** has a significant effect on the performances, as previous works have suggested.
- Focus on the trade-off between **accuracy** and **complexity** of our models considering the memory footprint, the computational resources, and the power demand of each architecture

Dataset

The dataset used in this work consists of 374 histopathological images obtained from H&E-stained biopsies subdivided into three classes:

- **Chronic Lymphocytic Leukemia** (CLL): a blood cancer that starts in the blood-forming cells of the bone marrow
- **Follicular Lymphoma** (FL): a slow-growing non-Hodgkin lymphoma that begins in the follicles
- **Mantle Cell Lymphoma**(MCL): an aggressive non-Hodgkin lymphoma that affects the mantle zone of lymphatic tissue

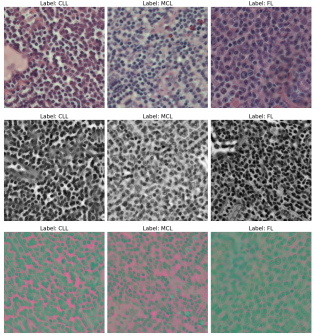
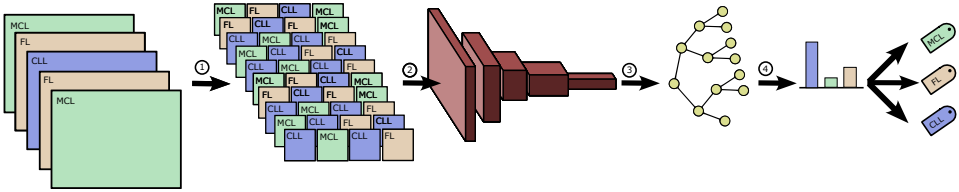
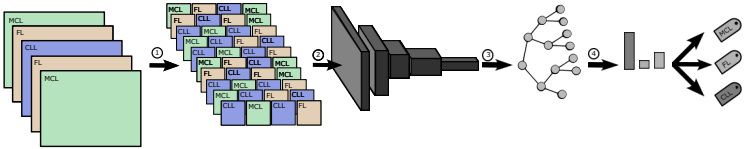


Figure: Image patches: RGB (top), gray (middle), LAB (bottom)

Processing Pipeline



① Preprocessing and patches extraction
② Features extraction via CNN
③ Classification with machine learning models
④ Label prediction with majority voting



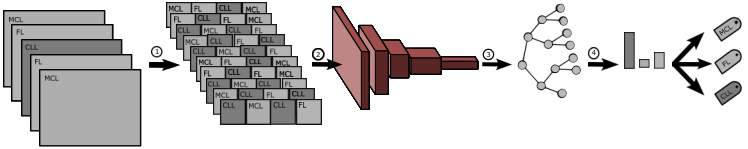
Images undergo the following transformations:

Color conversion: into target colorspace and normalization in $[0, 1]$ range

Patches extraction: each image is split into smaller patches with 50% of overlap

Data augmentation: rotation, translation, and flipping (applied to patches)

Learning Frameworks



We aim to study different types of networks to find the most **efficient**, that is the one with the highest accuracy and lowest complexity.

These networks are trained **end-to-end**, i.e. they learn how to correctly classify the pathces.

Only later these architectures are exploited as **feature extractors**, removing the last dense layer.

Inception V4 (2017)

This network can capture both fine-grained and high-level information in the image thanks to the **inception modules**: building blocks of the networks that perform multiple convolutional operations in parallel. Outputs of these operations are then concatenated and fed into the next layer of the network.

Figure: Inception V4 main schema

ResNet 50 (2015)

The key feature of the ResNet architecture is the use of **residual connections**: after each step the input is added to the output of a convolutional layer, allowing the network to learn the residual difference between input and output rather than the output itself. This prevents the appearance of the **vanishing gradients** problem allowing the network to be much more deeper.

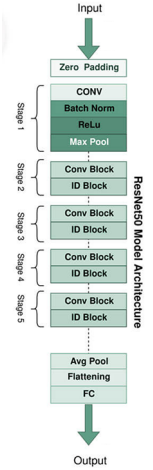


Figure: ResNet50 main schema

Shallow CNN

A **shallow** and **classical** CNN is implemented to evaluate the performances of a smaller and simpler network. The architecture comprises convolutional blocks (with batch normalization and ReLU function) that halve the input size at each step.

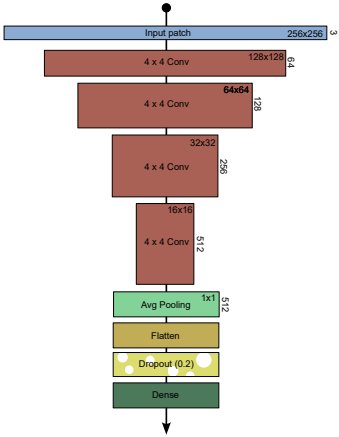


Figure: "Shallow" CNN schema

EfficientNet B0

EfficientNet B0 is the smaller network of a family of CNNs designed to achieve high accuracy while minimizing the number of parameters and computations required. This is achieved exploiting **compound scaling**, which balances the model depth, width, and resolution. Moreover, the network incorporates **depth-wise separable convolutions** and **squeeze-and-excitation** modules.

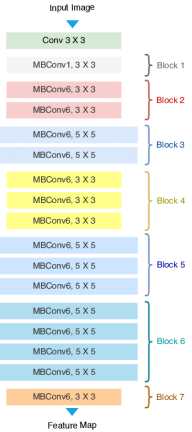
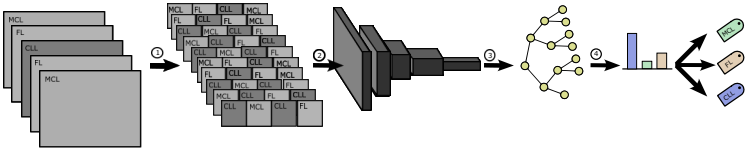


Figure: EffcientNet B0 main schema

Classifiers



In addition to neural networks only, two different frameworks are trained for the **classification task**

Those classifiers are **trained with features** extracted from the CNNs.

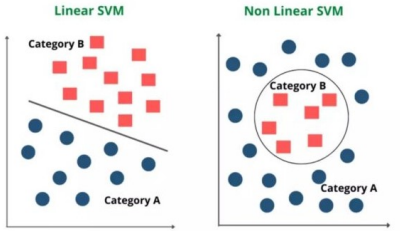
Models' hyper-parameters are tuned with **bayesian optimization** via Optuna

Support Vector Machine

Support Vector Machine (SVM) is a machine learning algorithm used for classification and regression tasks.

The algorithm seeks the **hyperplane** with the greatest margin between classes while minimizing the classification loss.

The algorithm can handle non-linearly separable data using the **kernel trick**, in which data is transformed into a higher dimensional space where a linear boundary can be found.



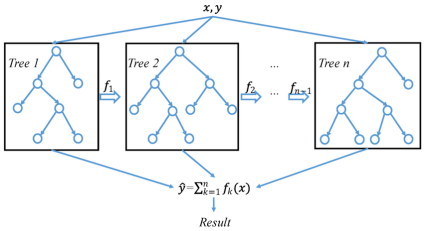
The parameters to be tuned are the kernel type, the regularization coefficient, and the kernel coefficient

XGBoost

XGBoost (eXtreme Gradient Boosting) is a ML algorithm that combines multiple weak models to form a robust and accurate model by iteratively correcting the mistakes of the previous models.

Decision trees are trained to predict the negative gradient of the loss function, while second-order gradient information captures **complex interactions** in the data.

Regularization, **feature pruning**, **weighting of instances** (and more) help to prevent overfitting.



There is a wide range of hyperparameters to be tuned, such as the learning rate, number of trees, tree depth, and regularization terms.

Results - Feature extractors training

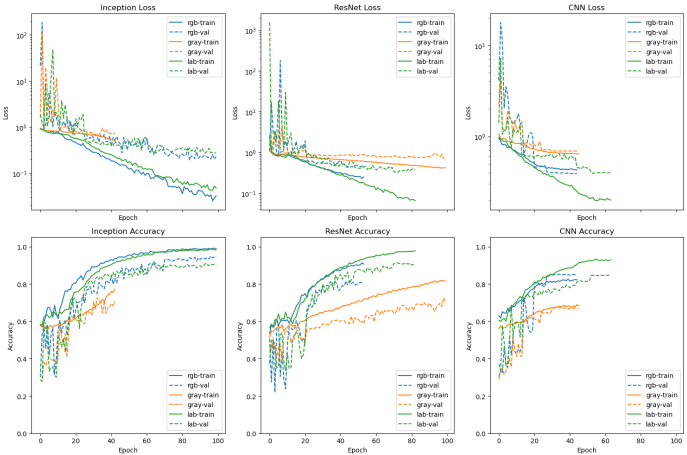


Figure: Loss (top row) and accuracy (bottom row) curves during training of the (from left to right) Inception, ResNet and CNN. The dotted lines show the validation behavior. The RGB data is in blue, the grayscale in orange, and the LAB in green

Results - Classification metrics

Model	Color space	Accuracy			Precision			Recall			F1-score			ROC AUC		
		Train	Val	Test	Train	Val	Test	Train	Val	Test	Train	Val	Test	Train	Val	Test
Inception	LAB	1	0.947	0.933	1	0.947	0.933	1	0.943	0.936	1	0.944	0.933	1	0.995	0.986
	RGB	0.996	1	**0.973**	0.996	1	0.972	0.995	1	**0.974**	0.996	1	**0.973**	1	1	**0.998**
Inception+SVM	LAB	1	0.947	0.947	1	0.947	0.946	1	0.943	0.948	1	0.944	0.946	1	0.994	0.985
	RGB	0.996	0.987	**0.973**	0.996	0.987	0.974	0.995	0.985	**0.974**	0.996	0.986	**0.973**	1	1	0.996
Inception+XGB	LAB	1	0.960	0.960	1	0.959	0.960	1	0.958	0.962	1	0.958	0.960	1	0.995	0.986
	RGB	0.996	0.973	**0.973**	0.996	0.976	0.972	0.995	0.970	**0.974**	0.996	0.972	**0.973**	1	1	0.997
ResNet	LAB	0.991	0.946	0.907	0.991	0.947	0.903	0.991	0.945	0.904	0.991	0.944	0.903	1	0.997	0.976
	RGB	0.893	0.867	0.893	0.888	0.864	0.892	0.888	0.866	0.889	0.888	0.864	0.889	0.979	0.977	0.977
ResNet+SVM	LAB	1	0.973	0.920	1	0.973	0.918	1	0.972	0.919	1	0.972	0.918	1	0.998	0.981
	RGB	0.915	0.867	0.88	0.912	0.872	0.878	0.912	0.870	0.877	0.912	0.866	0.877	0.988	0.981	0.979
ResNet+XGB	LAB	1	0.973	0.933	1	0.973	0.931	1	0.972	0.932	1	0.972	0.931	1	0.998	0.975
	RGB	0.888	0.893	0.893	0.884	0.893	0.892	0.882	0.892	0.889	0.882	0.891	0.889	0.973	0.980	0.970
CNN	LAB	0.982	0.867	0.907	0.982	0.867	0.907	0.982	0.867	0.908	0.982	0.865	0.906	0.999	0.978	0.973
	RGB	0.942	0.907	0.920	0.944	0.902	0.921	0.938	0.901	0.922	0.939	0.901	0.919	0.992	0.970	0.977
CNN+SVM	LAB	1	0.947	0.893	1	0.952	0.893	1	0.943	0.894	1	0.945	0.893	1	0.990	0.978
	RGB	1	0.920	0.933	1	0.917	0.933	1	0.913	0.936	1	0.914	0.933	1	0.981	0.975
CNN+XGB	LAB	1	0.907	0.933	1	0.905	0.936	1	0.903	0.937	1	0.903	0.933	1	0.988	0.971
	RGB	1	0.907	0.933	1	0.902	0.933	1	0.903	0.936	1	0.902	0.933	1	0.985	0.975
EfficientNet	RGB	0.964	0.813	0.920	0.963	0.964	0.964	0.816	0.821	0.814	0.922	0.924	0.920	0.996	0.965	0.967
EfficientNet + SVM	RGB	0.987	0.880	0.933	0.987	0.986	**0.987**	0.889	0.879	0.878	0.934	0.937	0.934	1.000	0.980	0.967
EfficientNet + XGB	RGB	0.955	0.867	0.907	0.956	0.955	0.955	0.871	0.869	0.867	0.907	0.910	0.907	0.998	0.970	0.965
Orlov et. al. (2010)		0.990	0.990	0.990												
Janowczyk et. al. (2016)		0.966	0.966	0.966												
Tambe et. al. (2019)		0.975	1	0.973												
Zeyad et. al. (2022)		1	1	1												

Results - Classification metrics

Model	Color space	Accuracy			Precision			Recall			F1-score			ROC AUC		
		Train	Val	Test	Train	Val	Test	Train	Val	Test	Train	Val	Test	Train	Val	Test
Inception	LAB	1	0.947	0.933	1	0.947	0.933	1	0.943	0.936	1	0.944	0.933	1	0.995	0.986
	RGB	0.996	1	**0.973**	0.996	1	0.972	0.995	1	**0.974**	0.996	1	**0.973**	1	1	**0.998**
Inception+SVM	LAB	1	0.947	0.947	1	0.947	0.946	1	0.943	0.948	1	0.944	0.946	1	0.994	0.985
	RGB	0.996	0.987	**0.973**	0.996	0.987	0.974	0.995	0.985	**0.974**	0.996	0.986	**0.973**	1	1	0.996
Inception+XGB	LAB	1	0.960	0.960	1	0.959	0.960	1	0.958	0.962	1	0.958	0.960	1	0.995	0.986
	RGB	0.996	0.973	**0.973**	0.996	0.976	0.972	0.995	0.970	**0.974**	0.996	0.972	**0.973**	1	1	0.997
ResNet	LAB	0.991	0.946	0.907	0.991	0.947	0.903	0.991	0.945	0.904	0.991	0.944	0.903	1	0.997	0.976
	RGB	0.893	0.867	0.893	0.888	0.864	0.892	0.888	0.866	0.889	0.888	0.864	0.889	0.979	0.977	0.977
ResNet+SVM	LAB	1	0.973	0.920	1	0.973	0.918	1	0.972	0.919	1	0.972	0.918	1	0.998	0.981
	RGB	0.915	0.867	0.88	0.912	0.872	0.878	0.912	0.870	0.877	0.912	0.866	0.877	0.988	0.981	0.979
ResNet+XGB	LAB	1	0.973	0.933	1	0.973	0.931	1	0.972	0.932	1	0.972	0.931	1	0.998	0.975
	RGB	0.888	0.893	0.893	0.884	0.893	0.892	0.882	0.892	0.889	0.882	0.891	0.889	0.973	0.980	0.970
CNN	LAB	0.982	0.867	0.907	0.982	0.867	0.907	0.982	0.867	0.908	0.982	0.865	0.906	0.999	0.978	0.973
	RGB	0.942	0.907	0.920	0.944	0.902	0.921	0.938	0.901	0.922	0.939	0.901	0.919	0.992	0.970	0.977
CNN+SVM	LAB	1	0.947	0.893	1	0.952	0.893	1	0.943	0.894	1	0.945	0.893	1	0.990	0.978
	RGB	1	0.920	0.933	1	0.917	0.933	1	0.913	0.936	1	0.914	0.933	1	0.981	0.975
CNN+XGB	LAB	1	0.907	0.933	1	0.905	0.936	1	0.903	0.937	1	0.903	0.933	1	0.988	0.971
	RGB	1	0.907	0.933	1	0.902	0.933	1	0.903	0.936	1	0.902	0.933	1	0.985	0.975
EfficientNet	RGB	0.964	0.813	0.920	0.963	0.964	0.964	0.816	0.821	0.814	0.922	0.924	0.920	0.996	0.965	0.967
EfficientNet + SVM	RGB	0.987	0.880	0.933	0.987	0.986	**0.987**	0.889	0.879	0.878	0.934	0.937	0.934	1.000	0.980	0.967
EfficientNet + XGB	RGB	0.955	0.867	0.907	0.956	0.955	0.955	0.871	0.869	0.867	0.907	0.910	0.907	0.998	0.970	0.965
Orlov et. al. (2010)		0.990	0.990	0.990												
Janowczyk et. al. (2016)		0.966	0.966	0.966												
Tambe et. al. (2019)		0.975	1	0.973												
Zeyad et. al. (2022)		1	1	1												

Results - Classification metrics (2)

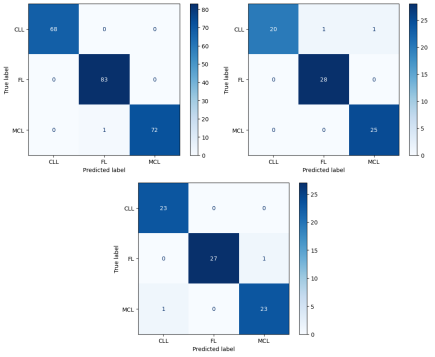


Figure: Confusion matrix for our Inception+XGB model. From top to bottom the values for the train, validation and test set respectively

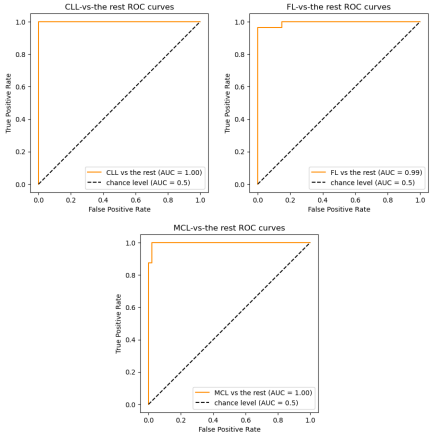


Figure: ROC curve for the 3 classes of the test set for our Inception+XGB model

Results - Performace efficiency trade-off

Architecture	Inception	ResNet	CNN	EfficientNet_B0
Parameters (Million)	41.2	23.5	2.7	4.1
Training Time (s/epoch)	120	90	70	60
Required epochs	100	65	50	100

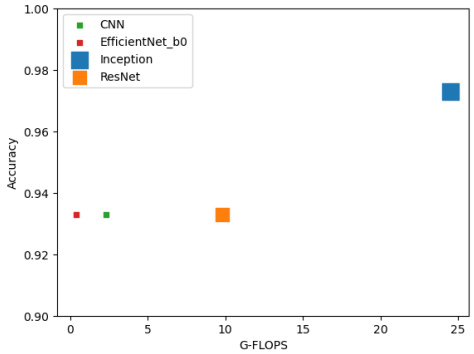


Figure: Accuracy vs G-FLOPS plot. The marker size is proportional to the number of parameters of the feature extractor

Final remarks

- With the exception of the grayscale case, no clear indication of which is the most suitable color space was found
- We were able to replicate previous results and come really close to state-of-the-art performance
- We found some promising indications regarding the performance-efficiency trade-off, suggesting the feasibility of **less memory and computationally demanding solutions**
- To better assess the true potential of neural networks for lymphoma classification **larger** and more **complex** datasets should be considered

Future works

- Investigate how changing the size and stride of the patches affects the model performance
- Tune the number of output features that are fed to the classifiers, eventually implement some **dimensionality reduction** techniques (e.g. PCA)
- Consider **model ensembling** or **feature fusion** to boost the performance of the smaller models
- Leverage domain knowledge to extract **hand-crafted features** and combine them with the one extracted by the CNNs

References

[Orlov2010] Orlov et al., "*Automatic classification of lymphoma images with transform-based global features*", IEEE Transactions on Information Technology in Biomedicine, 2010

[Janowczyk2016] Janowczyk et al., "*Deep learning for digital pathology image analysis: A comprehensive tutorial with selected use cases*", Journal of pathology informatics, 2016

[Tambe2019] Tambe et al., "*Towards designing an automated classification of lymphoma subtypes using deep neural networks*", Proceedings of the ACM India joint international conference on data science and management of data, 2019

[Zeyad2022] Zeyad et al., "*Diagnosis of Histopathological Images to Distinguish Types of Malignant Lymphomas Using Hybrid Techniques Based on Fusion Features*", 2022

Code available at https://github.com/lorenzo-saccaro/HDA-lymphoma-classification