Hierarchical Convolutional Neural Networks for Breast Cancer Detection

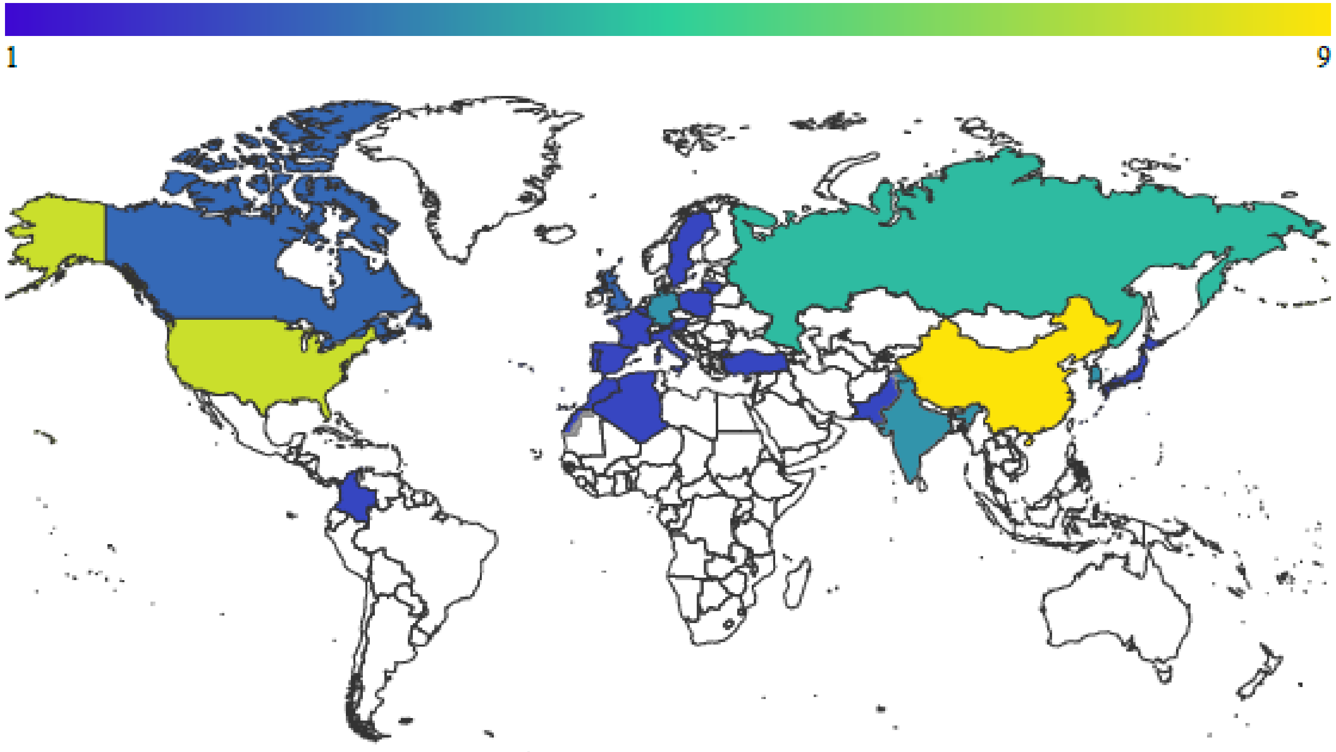
Ismaël Koné, Lahsen Boulmane

Faculté des Sciences – Université Moulay Ismail (UMI)

Meknès - Morocco

Context

- Challenge ICIAR 2018 (International Conference on Image Analysis and Recognition)
- Breast cancer detection (on BreAst Cancer Histology images -BACH)
- 51 teams worldwide



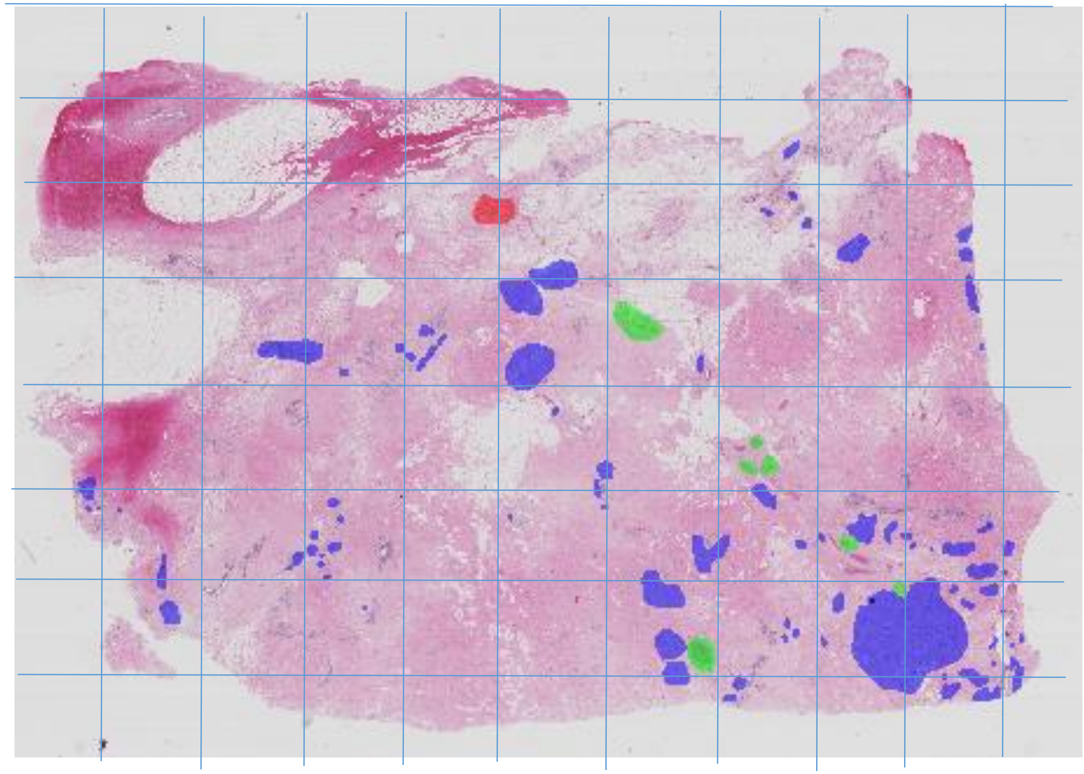
Distribution of ICIAR competitors

Problem

- Traditional approach through Breast Biopsy image analyses
- Very large Images 40K x 60K pixels for one patient (8GB in a numpy array).
- Patch-wise manual analysis by doctors
- Laborious , time consuming, error prone and sometimes subjective (among doctors)

Main objective of the competition

suggest a method to automatize this task

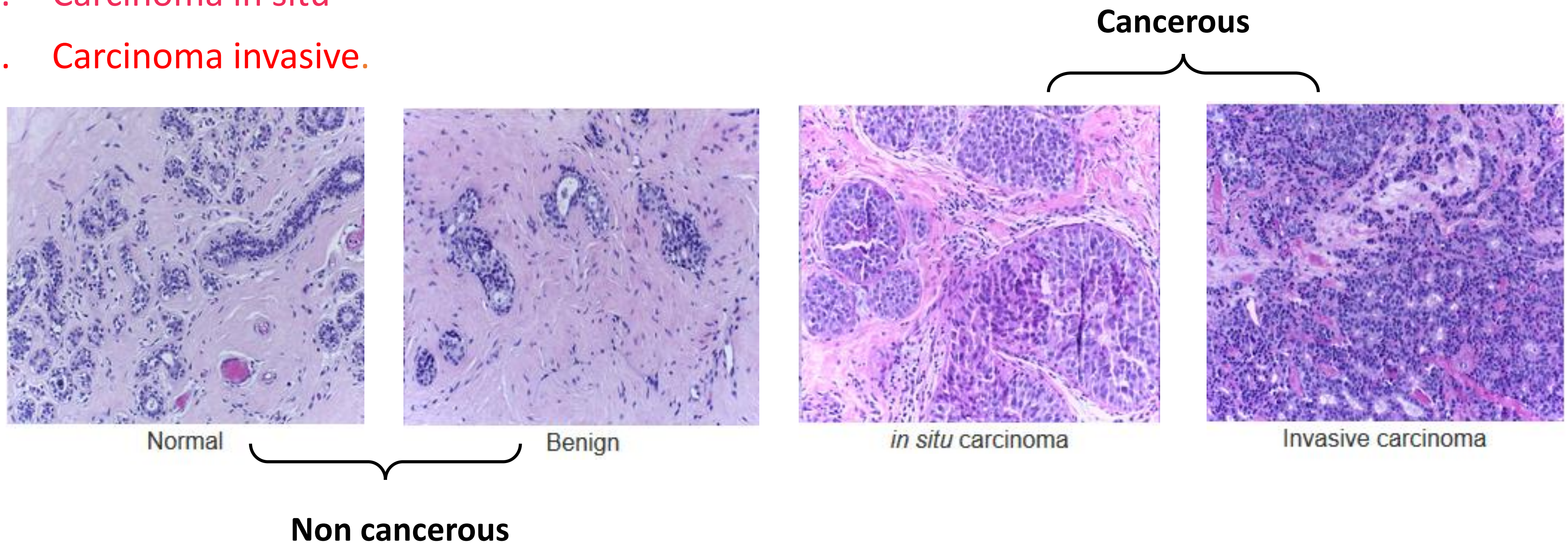


Breast Cancer Biopsy Image (40K x 60K pixels)

Specific objective

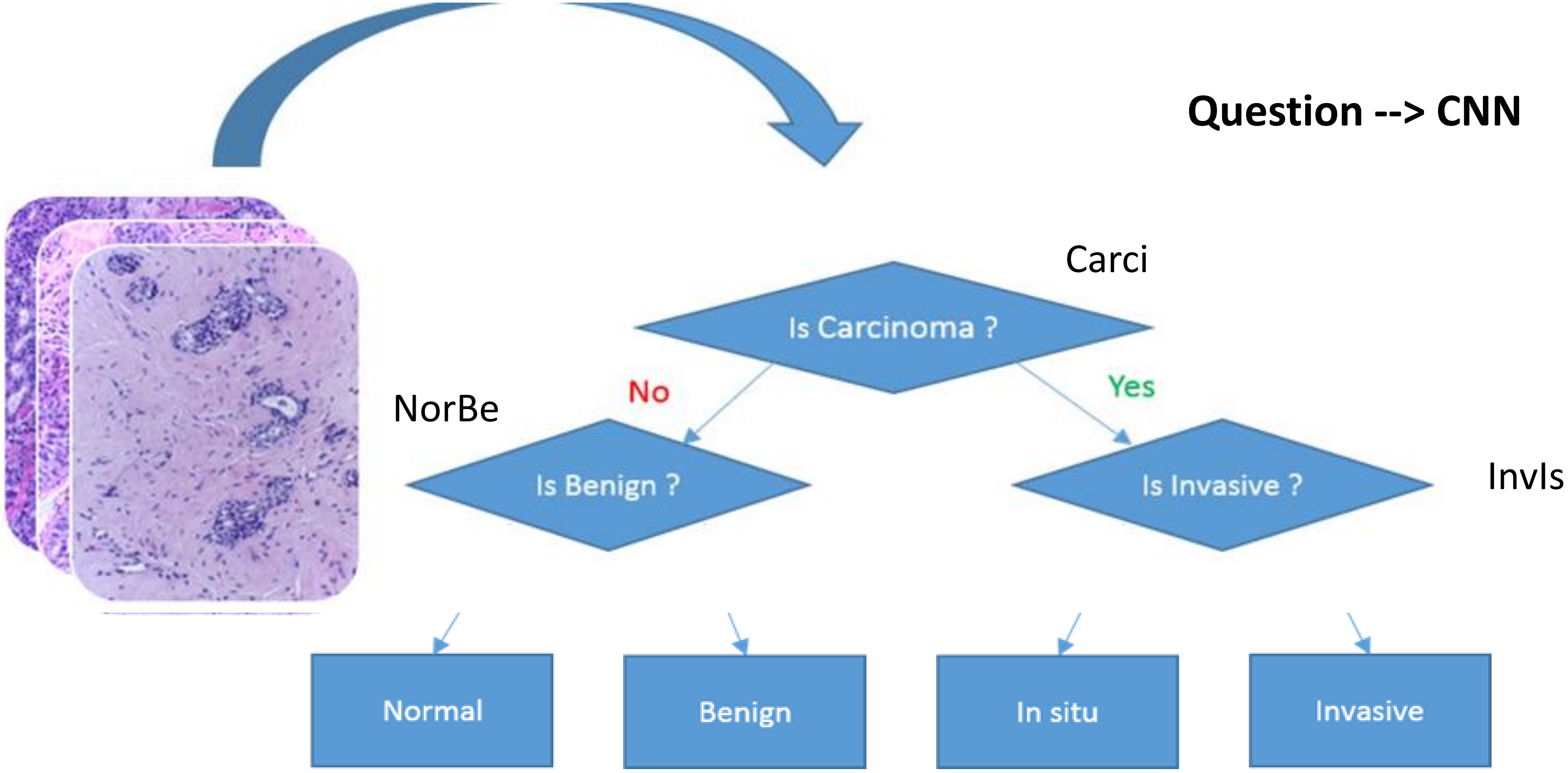
Image classification into 4 pathological groups:

1. Normal
2. Benign
3. Carcinoma in situ
4. Carcinoma invasive.



Our approach

At a high level, we divide the 4-categories classification problem into a hierarchy of simple binary classification problems.



What are these models ?

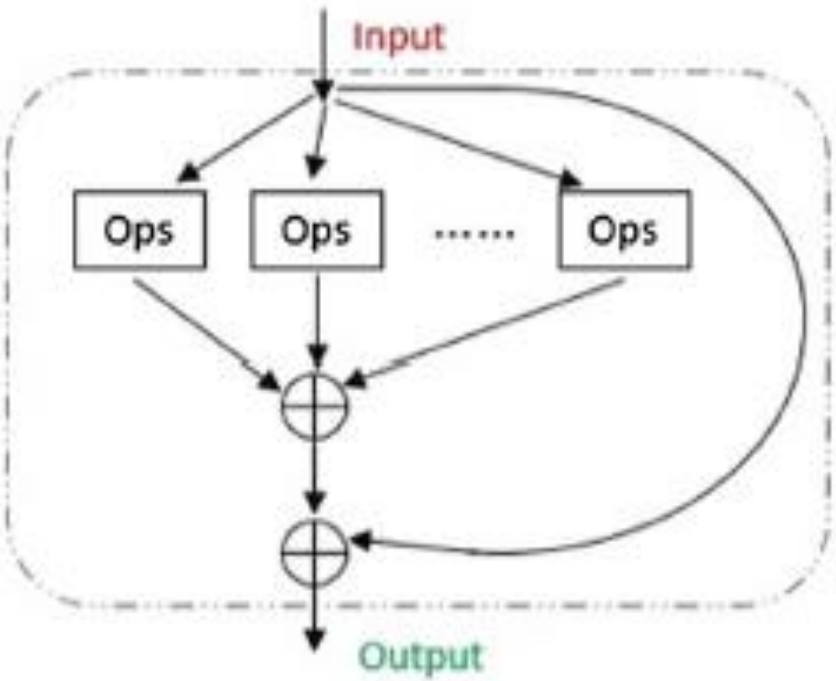
Each Convolutional Neural Network (CNN) $\longrightarrow$ ResNeXt model (Saining et al., CVPR 2017).

We used 3 ResNext50 (50 layers in depth) pretrained on ImageNet

We fine tuned each one to its corresponding binary classes of our hierarchical system.

What are these models ?

Ops = set of conv filters



(a)

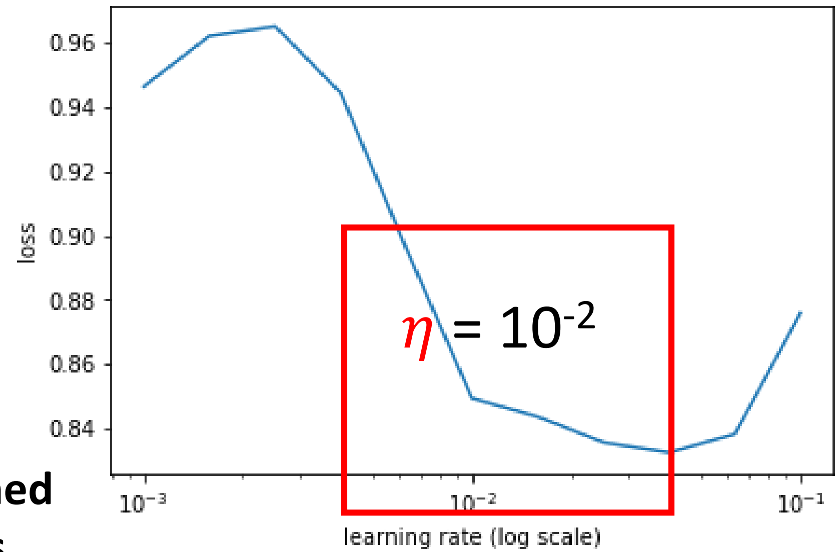
ResNeXt50 Base
2 x adaptive average pool 2D (1,1) concatenated
Batch Norm 1D + Dropout (Prob = 0.25)
Fully Connected layer (in=4096, out=512)
ReLU
Batch Norm 1D + Dropout (Prob = 0.5)
Fully Connected layer (in=512, out=2)
Softmax

(b)

How do we train the models?

Optimal learning rate η choice (Leslie,2017):

$$W = W - \eta \frac{\partial \mathcal{L}}{\partial \theta}$$



1. Train while increasing η from a small value
2. Plot the loss function against η
3. Choose η before loss explosion when still decreasing

ResNeXt50 Base
2 x adaptive average pool 2D (1,1) concatenated
Batch Norm 1D + Dropout (Prob = 0.25)
Fully Connected layer (in=4096, out=512)
ReLU
Batch Norm 1D + Dropout (Prob = 0.5)
Fully Connected layer (in=512, out=2)
Softmax

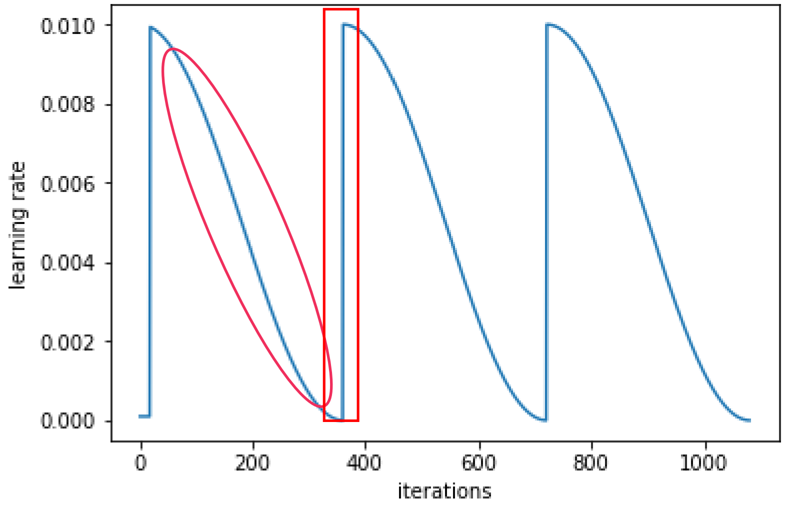
Pretrained Weights, frozen

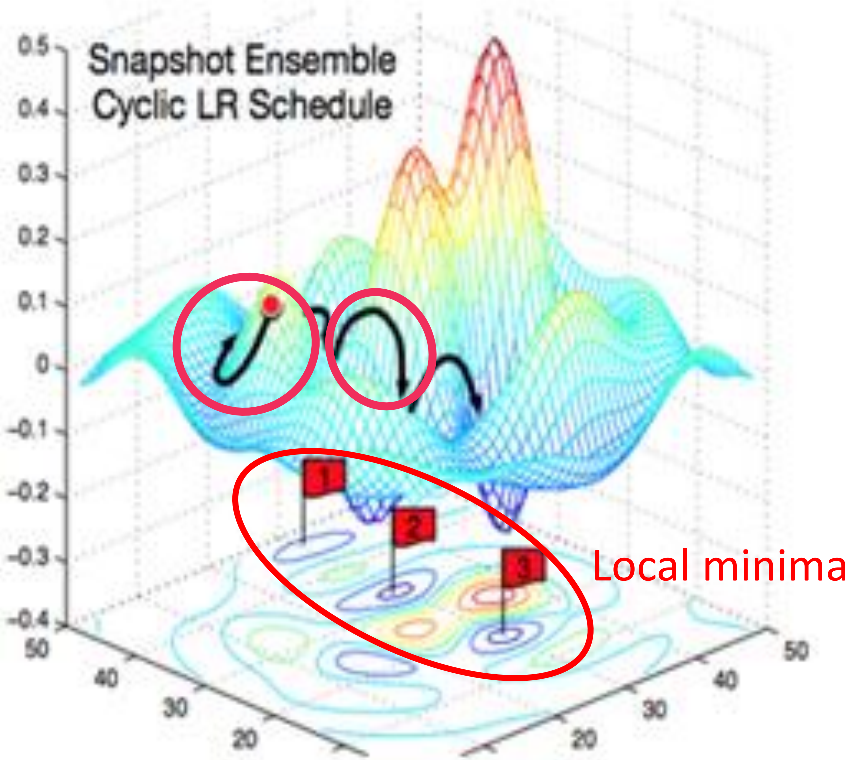
From Scratch using η

How do we train the models ?

- Learning Rate Scheduling

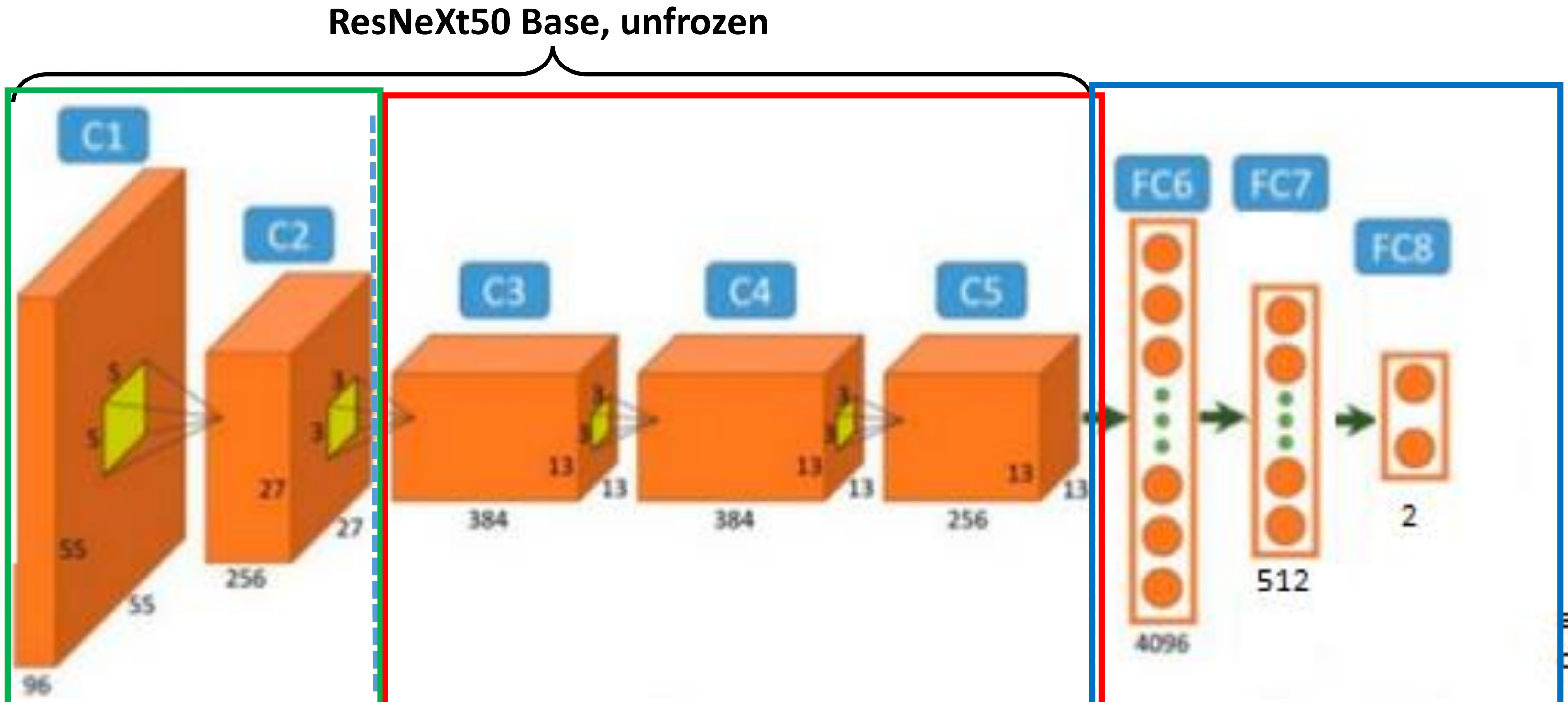
via Stochastic Gradient Descent with Warm Restarts (SGDR)[Loshchilov and Hutter, 2017].





How do we train the models ?

- Adaptation of the pretrained layers -Different learning rates to different set of layers (Howard et al.,2018)



adapted from Fast.ai course by J.Howard, 2017

First layers (Simple feats)	Middle layers (complex feats)	Last custom layers (high level feats)
$\eta/100$	$\eta/10$	η

Recap

1. Choose model
2. Train the model

   Optimize learning rate

   Schedule the learning rate

   Adapt the pretrained layers

Experiments

Dataset : 400 images (100/category), from BreAst Cancer Histology (BACH) images 2018 Grand Challenge.

Size: 2048 x 1536 pixels

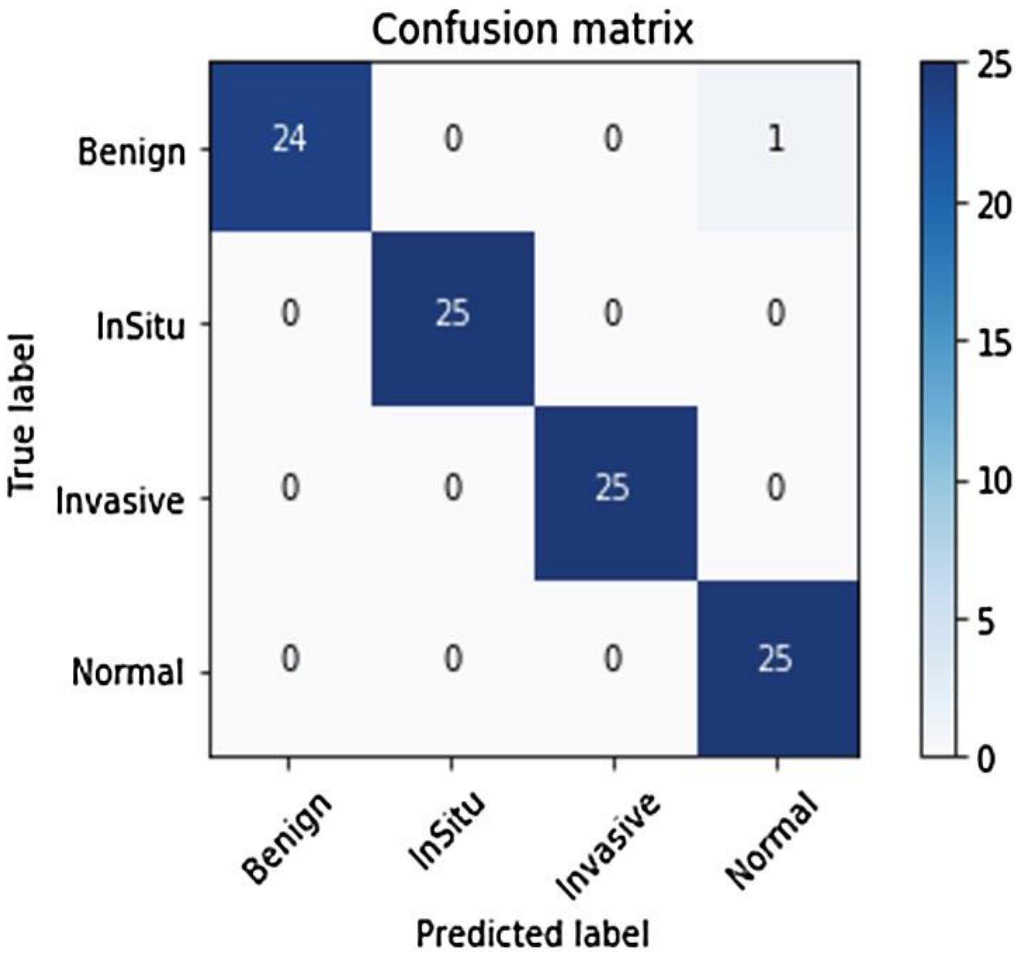
Split (75% vs 25%): 300 images for training set and 100 images for validation set.

Resize to 299 x 299 pixels

Data augmentation: Random rotations, flips, crops

Results

Models	Validation accuracy
Carci	100%
NorBe	98%
InvIs	100%
Whole system	99%



Competition results and Analysis

PART A - microscopy images

Position	Team	Part A	First Author	Last Author	Country
1	216	0,87	Sai Saketh Chennamsetty	Varghese Alex	India
1	248	0,87	Scotty Kwok		Hong Kong SAR
3	1	0,86	Nadia Brancati	Daniel Riccio	Italy
4	16	0,84	Bahram Marami	Jack Zeineh	USA
5	15	0,83	Xianfei Zheng	Yang Duan	China
5	54	0,83	Matthias Kohl	Maximilian Baust	United Kingdom|Germany
5	157	0,83	Yaqi Wang	Jiannan Fang	China
8	186	0,81	J. Steinfeldt	S. Jabari	
8	19	0,81	Ismael Kone	Lahsen Boulmane	Morocco
8	36	0,81	Imane Nedjar	Mohammed Amine Chikh	Algeria|Belgium
11	412	0,8	Kamalakkannan Ravi	Mohanasankar Sivaprakasam	India
11	94	0,8			
13	22	0,79	Zeya Wang	Eric P. Xing	USA
13	425	0,79	Hongliu CAO	Robert Sabourin	Canada|France
13	60	0,79	Kayoung Seo	Kyu-Hwan Jung	Korea
16	370	0,78	John-William Sidhom	Alexander S. Baras	USA
16	410	0,78	Yongxiang Huang		China
18	242	0,77	Yao Guo	Jun Liu	China
18	61	0,77	Nidhi Ranjan	A. D. Dileep	India
18	73	0,77	Amirreza Mahbod	Chunliang Wang	Austria|Sweden
21	18	0,76	Carlos A. Ferreira	Pedro Costa	Portugal
21	256	0,76	Gleb Makarchuk	Mikhail Belyaev	Russia
23	358	0,75	Mohammad Ibrahim Sarker	Dinar Akhmetzanov	South Korea|Russia
24	98	0,74	Alexander Rakhlin	Alexandr A. Kalinin	Russia|USA
25	164	0,72	Tomas Iesmantas	Robertas Alzbutas	Lithuania
25	253	0,72	Xinpeng Xie	Linlin Shen	China
25	268	0,72	Nick Weiss	Andre Homeyer	Germany
28	6	0,71	Ruqayya Awan	Nasir Rajpoot	United Kingdom
29	62	0,7	Fengfeng Liang		

We ranked 8th out of 51 teams with 81% Accuracy.
Is the system overfitting
(validation 99% Vs Test 81%) ?

Perhaps. But our explanations are:

1. We have nearly not use validation set (100 images).
2. We resized all images from 2048 x 1536 to 399 ×299. Then center cropped a 299 square. Thus a 299 x 100 pixels thrown away !

More Analysis

		Normal		Benign		*In situ*		Invasive	
Team	Acc	Se.	Sp.	Se.	Sp.	Se.	Sp.	Se.	Sp.
216 [20]	0.87	0.96	0.88	0.8	0.96	0.84	1.0	0.88	0.99
248 [21]	0.87	0.96	0.93	0.72	0.96	0.88	0.97	0.92	0.96
1 [22]	0.86	0.96	0.91	0.68	0.97	0.84	0.99	0.96	0.95
16 [23]	0.84	0.92	0.95	0.64	0.96	0.84	0.99	0.96	0.89
54 [25]	0.83	0.96	0.92	0.52	0.97	0.88	0.92	0.96	0.96
157 [26]	0.83	0.96	0.91	0.64	0.99	0.92	0.91	0.8	0.97
186	0.81	0.96	0.92	0.68	0.96	0.76	0.95	0.84	0.92
19 [27]	0.81	1.0	0.95	0.4	0.99	0.92	0.92	0.92	0.89
36	0.81	0.88	0.92	0.6	0.96	0.88	0.95	0.88	0.92
412	0.8	0.92	0.96	0.48	0.97	0.84	0.92	0.96	0.88
VGG16	0.58	0.84	0.84	0.64	0.84	0.72	0.87	0.36	0.97
Inception V3	0.77	0.92	0.93	0.44	0.96	0.88	0.87	0.84	0.93
ResNet 50	0.76	0.88	0.92	0.52	0.95	0.8	0.87	0.84	0.95
DenseNet 169	0.79	0.92	0.96	0.36	0.99	0.92	0.83	0.96	0.95

Sensitivity: Probability of detecting the pathology when it is present.
Specificity: Probability of ***not*** detecting the pathology when it is ***not*** present.

Our method (team 19) has the best specificity (0.99) and the worse sensivity (0.4) for the Benign class !

More Analysis

		Normal		Benign		*In situ*		Invasive	
Team	Acc	Se.	Sp.	Se.	Sp.	Se.	Sp.	Se.	Sp.
216 [20]	0.87	0.96	0.88	0.8	0.96	0.84	1.0	0.88	0.99
248 [21]	0.87	0.96	0.93	0.72	0.96	0.88	0.97	0.92	0.96
1 [22]	0.86	0.96	0.91	0.68	0.97	0.84	0.99	0.96	0.95
16 [23]	0.84	0.92	0.95	0.64	0.96	0.84	0.99	0.96	0.89
54 [25]	0.83	0.96	0.92	0.52	0.97	0.88	0.92	0.96	0.96
157 [26]	0.83	0.96	0.91	0.64	0.99	0.92	0.91	0.8	0.97
186	0.81	0.96	0.92	0.68	0.96	0.76	0.95	0.84	0.92
19 [27]	0.81	1.0	0.95	0.4	0.99	0.92	0.92	0.92	0.89
36	0.81	0.88	0.92	0.6	0.96	0.88	0.95	0.88	0.92
412	0.8	0.92	0.96	0.48	0.97	0.84	0.92	0.96	0.88
VGG16	0.58	0.84	0.84	0.64	0.84	0.72	0.87	0.36	0.97
Inception V3	0.77	0.92	0.93	0.44	0.96	0.88	0.87	0.84	0.93
ResNet 50	0.76	0.88	0.92	0.52	0.95	0.8	0.87	0.84	0.95
DenseNet 169	0.79	0.92	0.96	0.36	0.99	0.92	0.83	0.96	0.95

Sensitivity: Probability of detecting the pathology when it is present.
Specificity: Probability of ***not*** detecting the pathology when it is ***not*** present.

Our method (team 19) has the best specificity (0.99) and the worse sensivity (0.4) for the Benign class !

Lessons learned

1. Look closely at your dataset and reflect deeply on your problem/task. Perhaps a simple function can do your work no need to a ML model at all!
2. Analyze your cross validation results and use more metrics to get insights, i.e ROC, AUC, ….
3. Take initiative: If your tool doesn't give you what you want, it's probably an opportunity to make a contribution.

A code snippet to save the best model during training

Part 1

iskode Ismaël Koné

Hi everyone,
I'm sharing with you a small code that saves the best model after all epochs of a training run.
Below is a typical screen shot while calling the fit method.

```
currently support widgets.
[ 0.        0.09336  0.15755  0.94     ]
[ 1.        0.12014  0.10522  0.94     ]
[ 2.        0.11589  0.15387  0.95     ]
[ 3.        0.10822  0.06515  0.98     ]
[ 4.        0.10212  0.1072   0.96     ]
[ 5.        0.07325  0.09247  0.97     ]
[ 6.        0.06461  0.09069  0.97     ]
[ 7.        0.07976  0.14336  0.93     ]
[ 8.        0.06767  0.04317  0.99     ]
[ 9.        0.09978  0.0704   0.95     ]
[ 10.        0.08654   0.07103   0.98     ]
[ 11.        0.08053   0.11195   0.95     ]
[ 12.        0.08564   0.11092   0.97     ]
[ 13.        0.1457    0.08684   0.97     ]
```

```
class SaveBestModel(LossRecorder):
    def __init__(self, model, lr, name='best_model'):
        super().__init__(model.get_layer_opt(lr, None))
        self.name = name
        self.model = model
        self.best_loss = None
        self.best_acc = None

    def on_epoch_end(self, metrics):
        super().on_epoch_end(metrics)
        loss, acc = metrics
        if self.best_acc == None or acc > self.best_acc:
            self.best_acc = acc
            self.best_loss = loss
            self.model.save(f'{self.name}')
        elif acc == self.best_acc and  loss < self.best_loss:
            self.best_loss = loss
            self.model.save(f'{self.name}')
```

References

[1] Saining, X., Ross, G., Piotr, D., Zhuowen, T., Kaiming, H.: Aggregated residual transformations for deep neural networks. In: The IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp. 1492–1500 (2017).

[2] Leslie, N.: Cyclical learning rates for training neural networks. In: 2017 IEEE Winter Conference on Applications of Computer Vision (WACV), pp. 464–472 (2017). https://doi.org/10.1109/WACV.2017.58

[3] Howard, J.: Lesson 2: Deep learning v2. practical deep learning for coders (2018)

[4] Loshchilov, I., Hutter, F.: SGDR: Stochastic gradient descent with warm restarts. In: 6th International Conference on Learning Representations (ICLR) (2017)

Thank you so much for your attention !