# Deep Learning
# Practical Session 6

François Fleuret

https://fleuret.org/dlc/

February 8, 2021

## Introduction

The objective of this session is to observe the impact of residual connections and batch-normalization on the gradient norm at different depth in a residual network.

You can start this session with an embryo of code that includes an implementation of a residual network and an example of graph drawing with Matplotlib:

https://fleuret.org/dlc/src/dlc_practical_6_embryo.py

## 1 Modification of the ResNet implementation

Edit the implementation of the ResNet and ResNetBlock so that you can pass two Boolean flags `skip_connections` and `batch_normalization` to specify if these features are activated or not.

## 2 Monitoring the gradient norm

Write a function `get_stats(skip_connections, batch_normalization)` that

1. creates a model with 30 residual blocks, 10 channels, $3 \times 3$ kernels,
2. computes the norm of the gradient of the cross-entropy with respect to the weights of the first convolutional layer of each residual block, on 100 individual samples,
3. returns the $30 \times 100$ resulting tensor.

**Hint:** You can create a list of the weight tensors of the first convolution layer of each block with:

```
monitored_parameters = [ b.conv1.weight for b in model.resnet_blocks ]
```

and use it to get the gradient norm for each.

## 3 Graph

Plot for the four configurations of the two Boolean flags `skip_connections` and `batch_normalization` the average of the gradient norm vs. depth.

If you use a notebook, you can set the Maplotlib backend to the 'inline' one to have graphs appear in it with

```
%matplotlib inline
```