Computer Vision : Project

In the name of Deep Learning

Faustine Ginoux - Guillaume Lamine

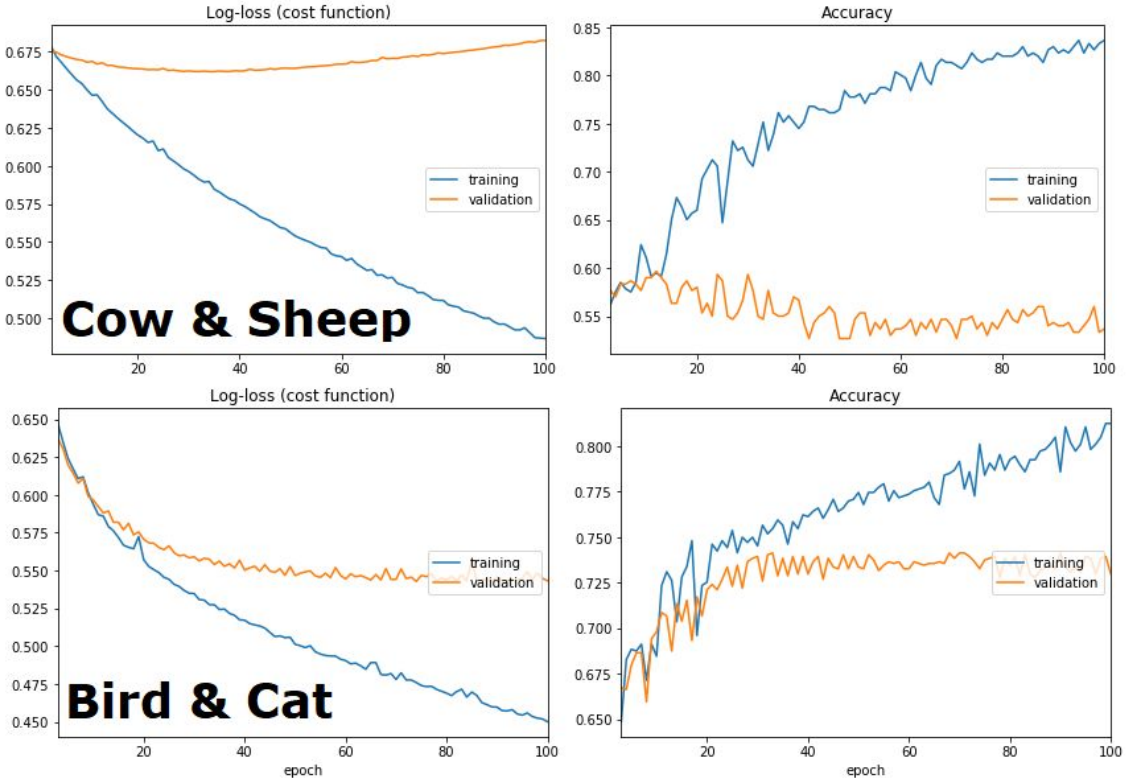
The dataset choice

Boats and Cats resized 128x128 pixels

The dataset “no-choice”

Low classification accuracy

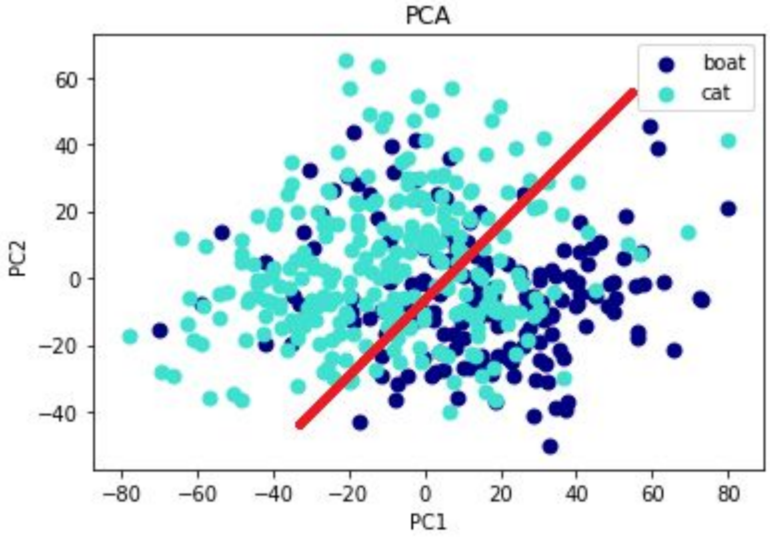
Boat and Cat results showed later (no spoilers!)

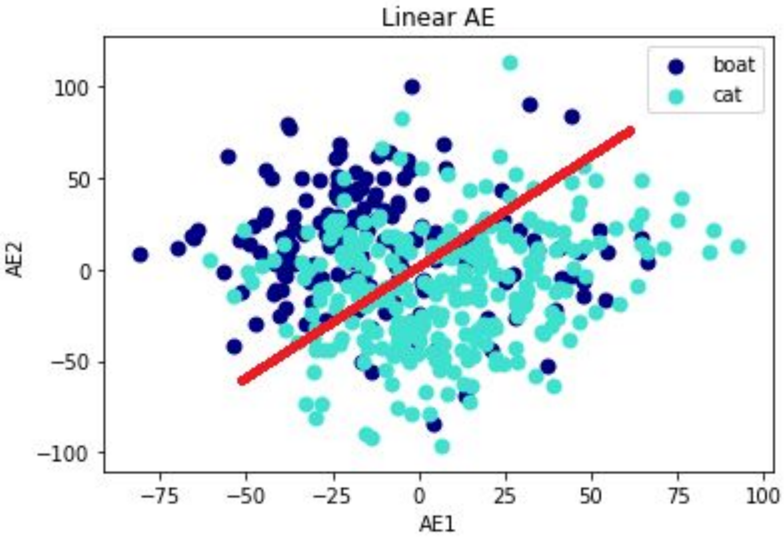


Autoencoders vs. PCA

Same subspace spanned but weights are different

Example in 2 dimensions:





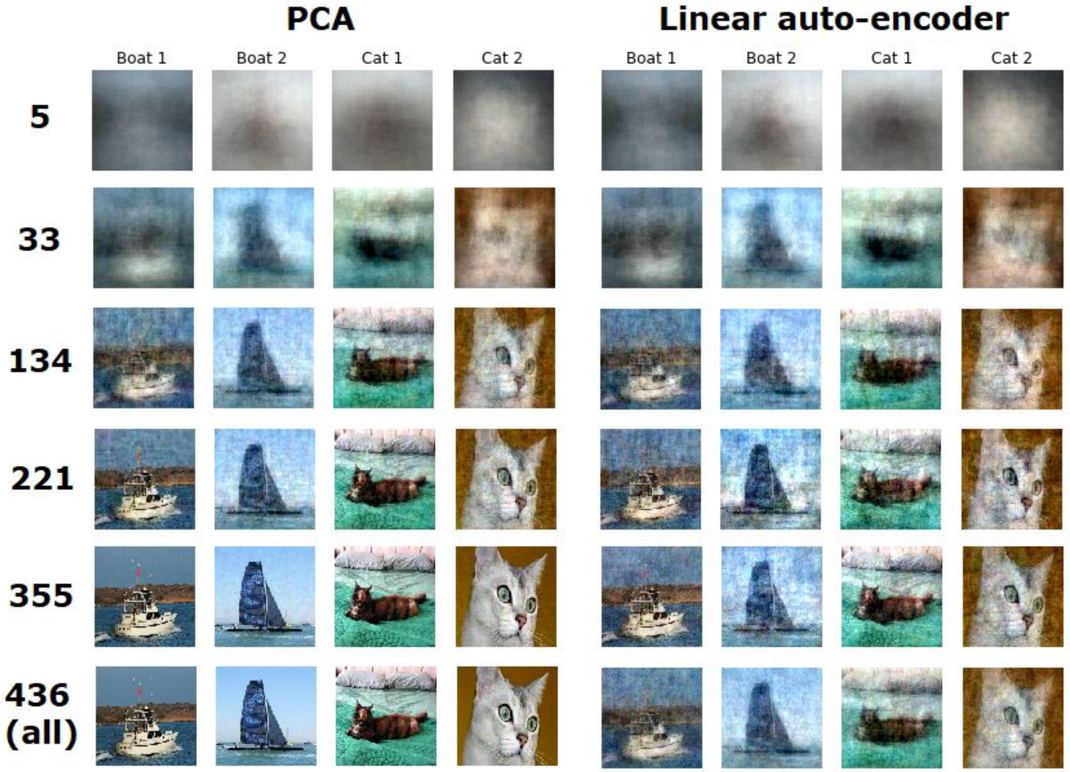
Autoencoders vs. PCA

Reconstruction

Loss function:

mean-squared error

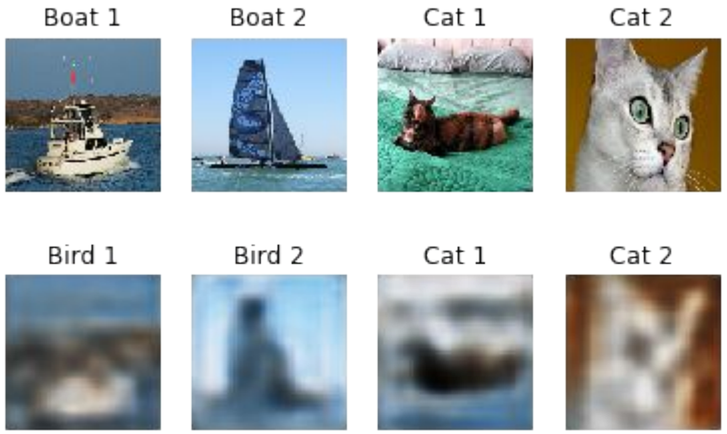
PCs / neurons	PCA error	LAE error
5	3.58	3.64
33	1.80	1.96
134	0.73	1.14
221	0.36	1.26
255	0.07	0.84
436	$10^{-13} \approx 0$	0.79



Non-Linear Convolutional Autoencoders

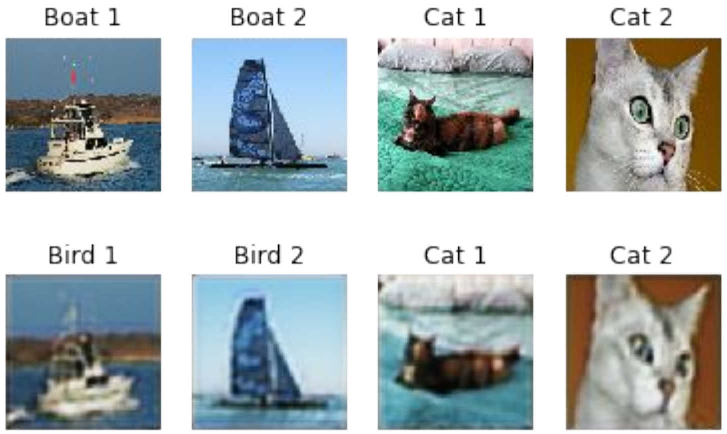
Reconstruction - different number of encoding variables

512



Error = **1.56**

8192



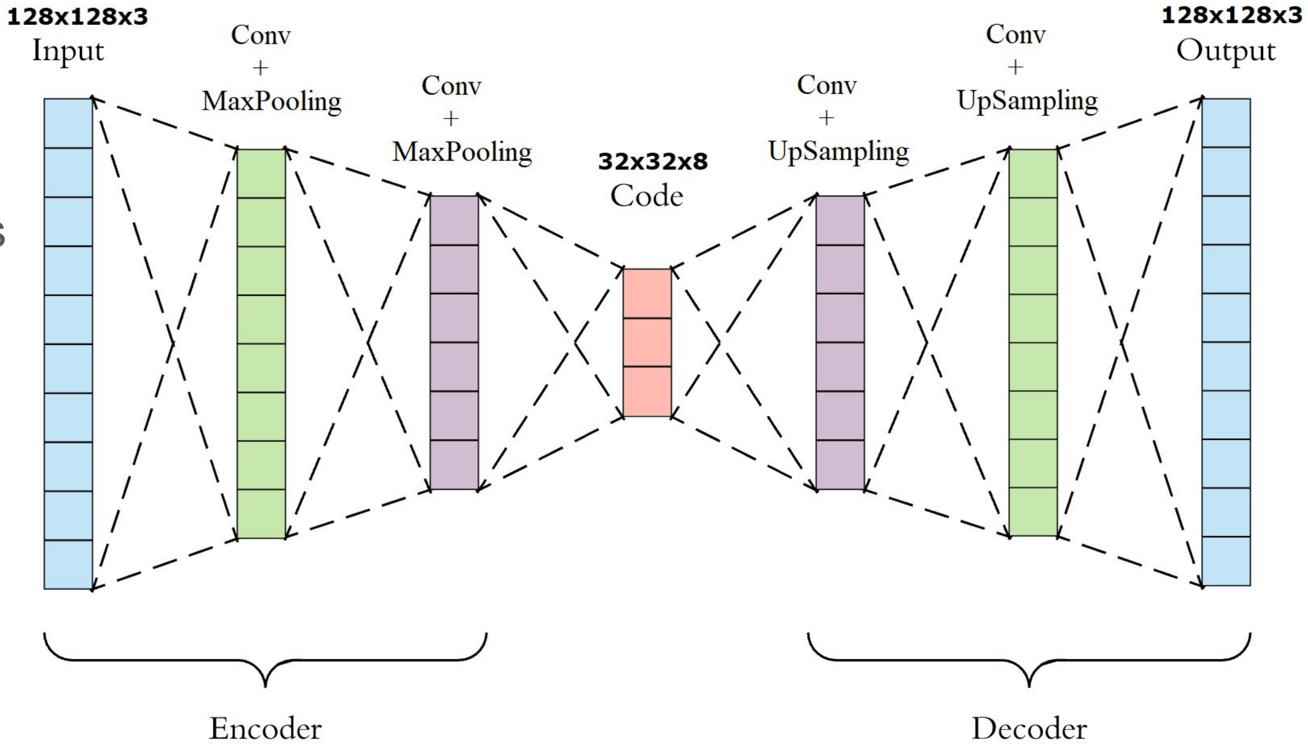
Error = **0.74** - Better than with linear auto-encoder

Loss function: **mean-squared error**

Classification task: use code of auto-encoder

1. Train the previous auto-encoder

   → 8192 coding variables



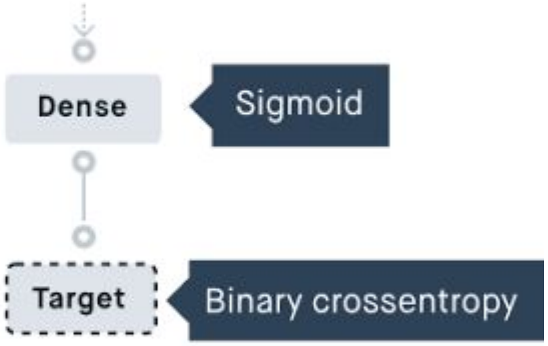
Classification task: multi-label classification

1. *Train the previous auto-encoder*

2. Take the coding variables and train a classifier

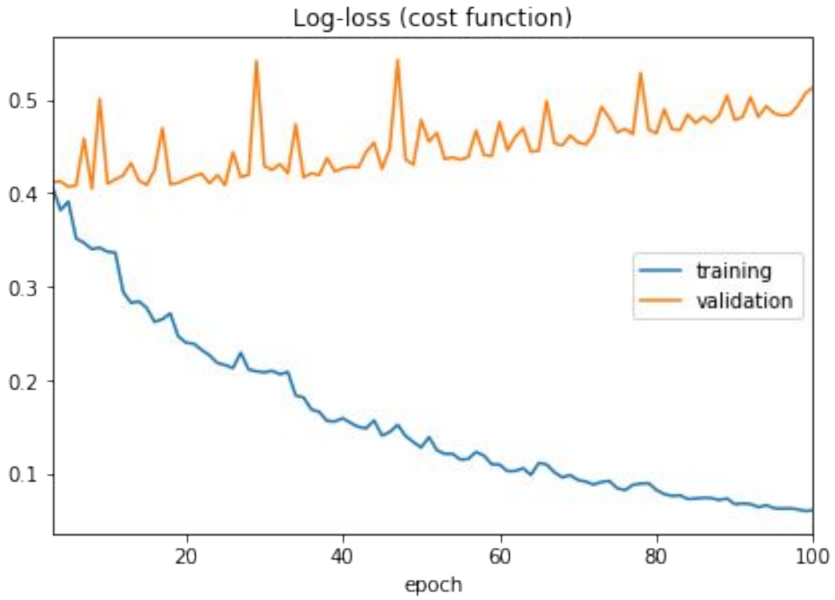
→ Flatten code

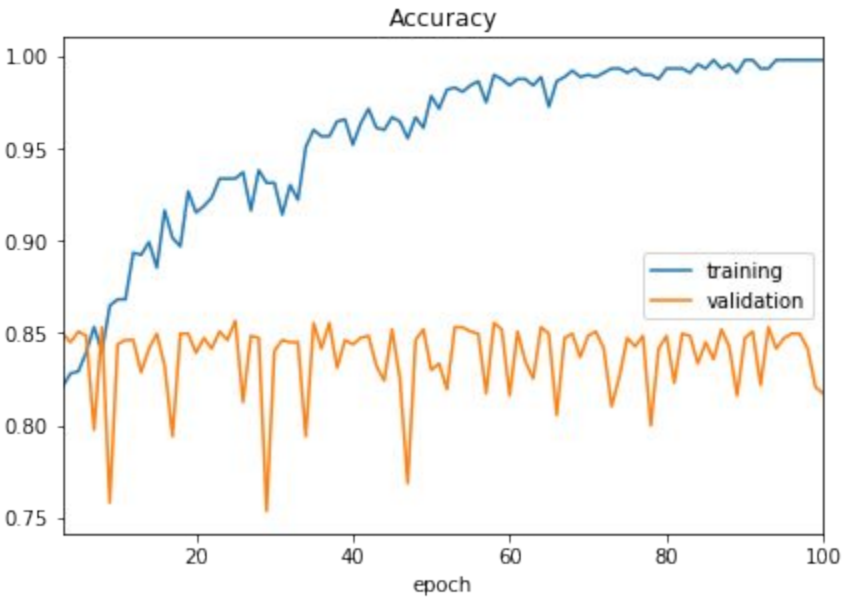
→ Dense layer with **2 outputs**



Classification task: results

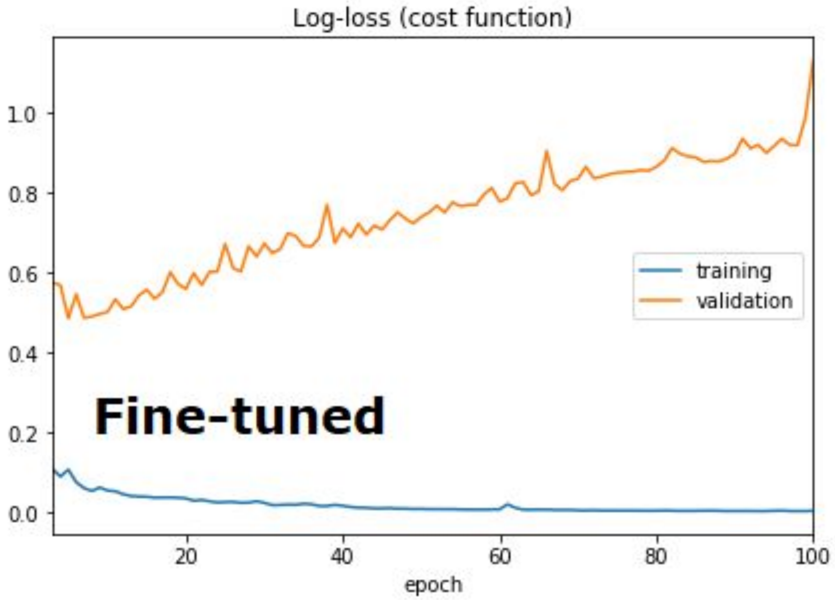
Overfitting

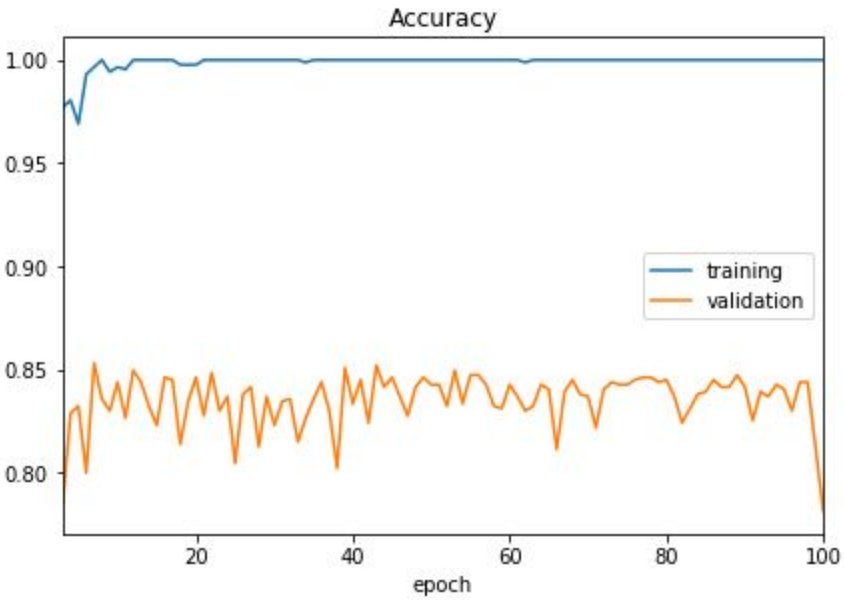




Classification task: fine-tuning

No improvement on validation set

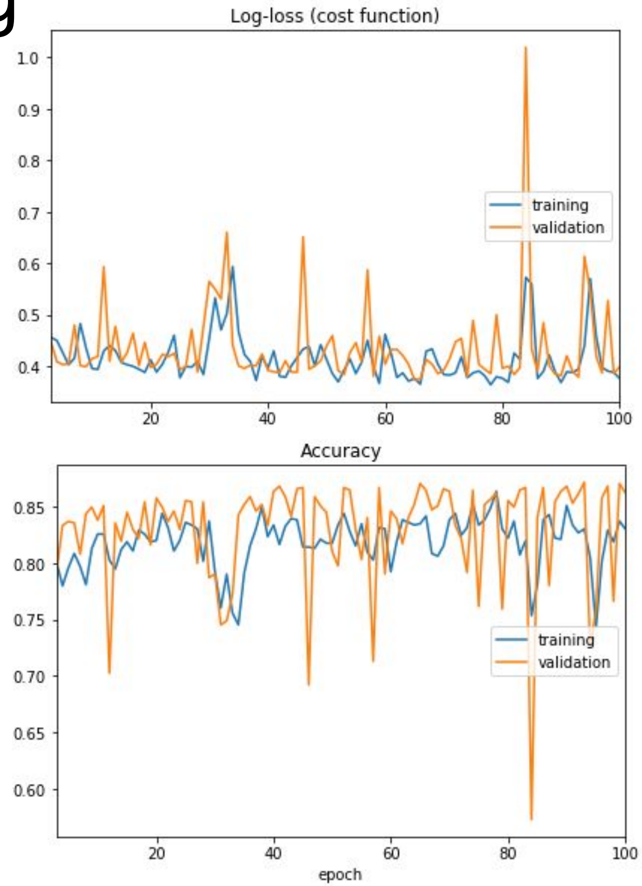




Classification task: avoid overfitting

- Greedy layer-wise training of auto-encoder
- Deep CNNs trained from scratch
- Activity regularization: L1, L2, dropout
- More training data
- Fewer coding variables
- Data augmentation

Data augmentation



Classification task: conclusion

- Classification on validation set: 85% - better than random!

- 41% of the boats misclassified
- 23% of the cats misclassified

→ Use data augmentation only on boats to obtain a balanced dataset

→ Select only good pictures of boats

- Try pre-trained network

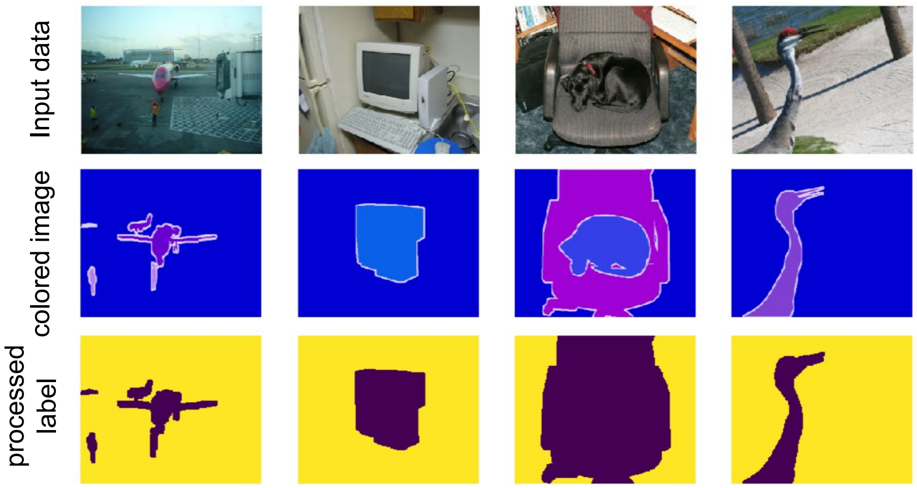
Segmentation task

The task : classification for each pixel

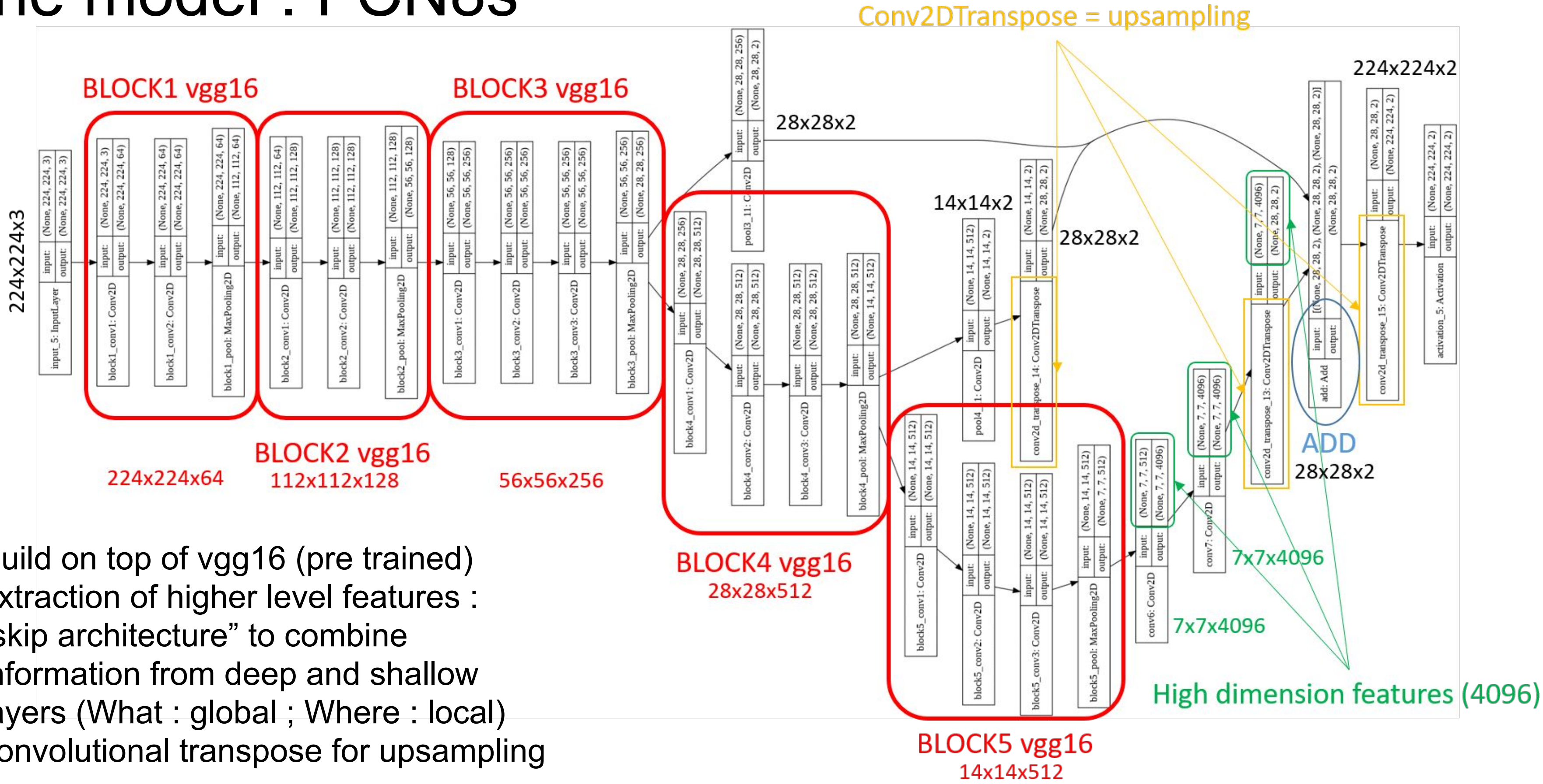
All the dataset is taken for better performances

The data processing :

- binary segmentation
- one hot encoding
- extract labels from colored images
- dimensions : 224 x 224 (for vgg16)



The model : FCN8s



- Build on top of vgg16 (pre trained)
- extraction of higher level features : "skip architecture" to combine information from deep and shallow layers (What : global ; Where : local)
- convolutional transpose for upsampling

Performance metrics for segmentation

We want a criteria that reflects a “good” segmentation

- pixel accuracy : risk of bias towards background

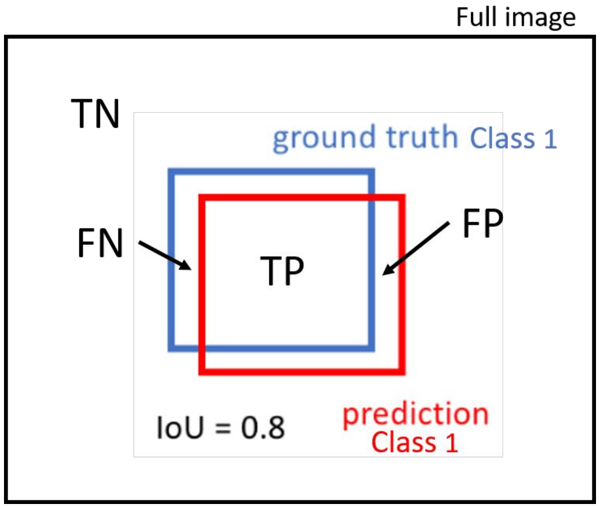
$$Acc = \frac{TP + TN}{TP + TN + FP + FN}$$

Better : Dice score and Intersection over Union (IoU)

- both measures of similarity between sets

$$IoU = \frac{|X \cap Y|}{|X \cup Y|} = \frac{TP}{TP + FP + FN}$$

$$Dice = \frac{2 \cdot |X \cap Y|}{|X| + |Y|} = \frac{2 \cdot TP}{2 \cdot TP + FP + FN}$$



Training and results

Training still with categorical cross-entropy (maximize log likelihood)

Dice and IoU are not differentiable

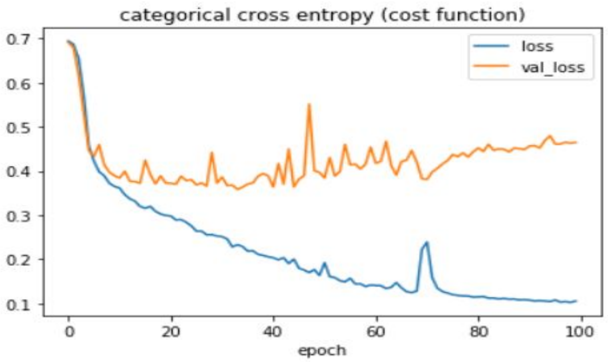
Comments:

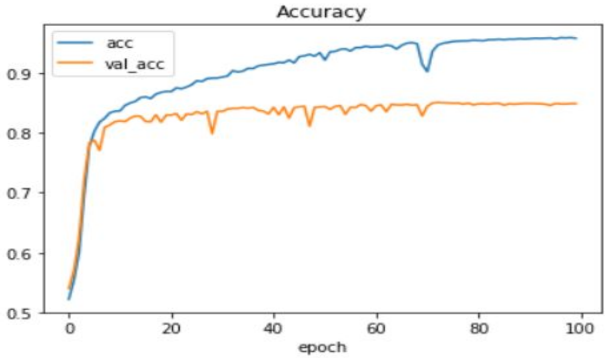
- loss: overfitting around 30 epochs
- Accuracy is still rising

Metrics	50 epochs	75 epochs	100 epochs	200 epochs
IoU_0	0.580	0.812	0.811	0.809
IoU_1	0.779	0.574	0.566	0.568
$Dice_0$	0.735	0.896	0.896	0.895
$Dice_1$	0.876	0.729	0.723	0.724
mean Dice score	0.805	0.813	0.809	0.809
mean IoU	0.680	0.693	0.689	0.688

Class 0 : background Class 1 : objects

Model	2012 PASCAL VOC (mIoU)
FCN	62.2
ParseNet	69.8
Conv & Deconv	72.5
FPN	X
PSPNet	85.4
Mask R-CNN	X
DeepLab	79.7
DeepLabv3	86.9
DeepLabv3+	89.0
PANet	X
EncNet	85.9





https://medium.com/@arthur_ouaknine/review-of-deep-learning-algorithms-for-image-semantic-segmentation-509a600f7b57

Visual evaluation of segmentation

- Visually, seems to get more refined BUT
- best dice and IoU at 75 epochs
- As supposed, where accuracy gets better, tendency to diminish the surface of object classes

- squared board effect, maybe due to transpose convolutions

Segmentation task : conclusion

The results are in the range of the results of the literature : good !

To go further :

- Optimize directly Dice or IoU (approximation functions)
- Generalize task to multi-class : clustering to extract all 22 colors ?
- Implement better architecture : U-net ?
- Search reason for squared effect in predictions