Réseaux de neurones profonds

Guillaume Bourmaud

PLAN

I. CNN profond

II. Modèles de fondation

I) CNN profond

I)

Rappel des ingrédients du « Deep Learning »

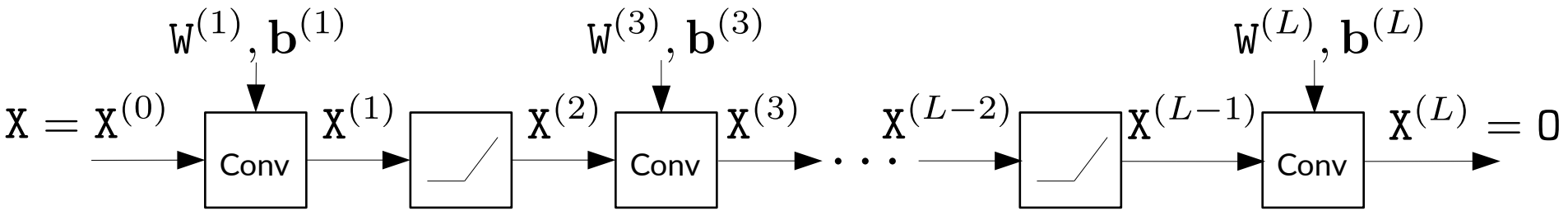
1) Grande base de données étiquetées

2) Grande capacité de calculs en parallèle (GPU)

3) « Bonne » architecture de réseau de neurones profond

I)

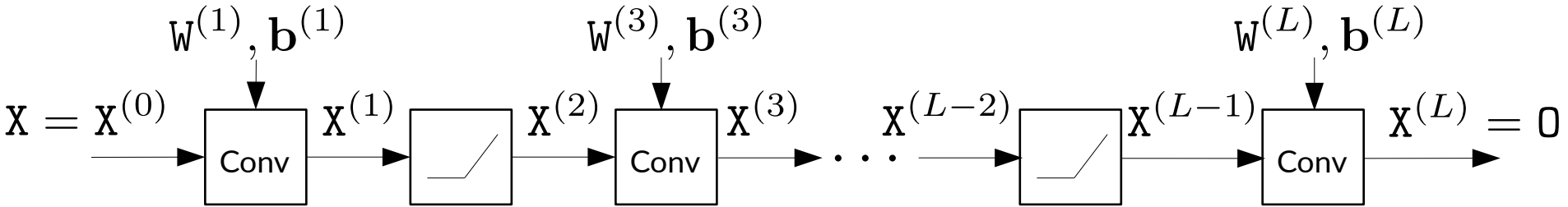
Limites du CNN « classique »

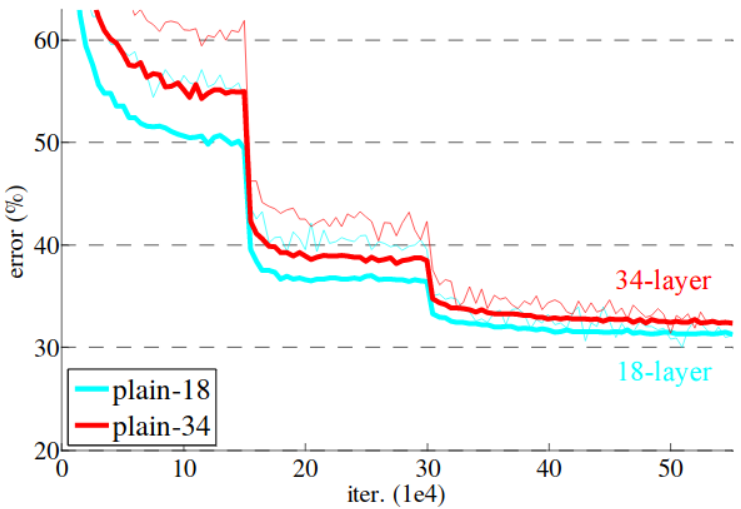


He, Kaiming, et al. "Deep residual learning for image recognition." Proceedings of the IEEE conference on computer vision and pattern recognition. 2016.

I)

Limites du CNN « classique »

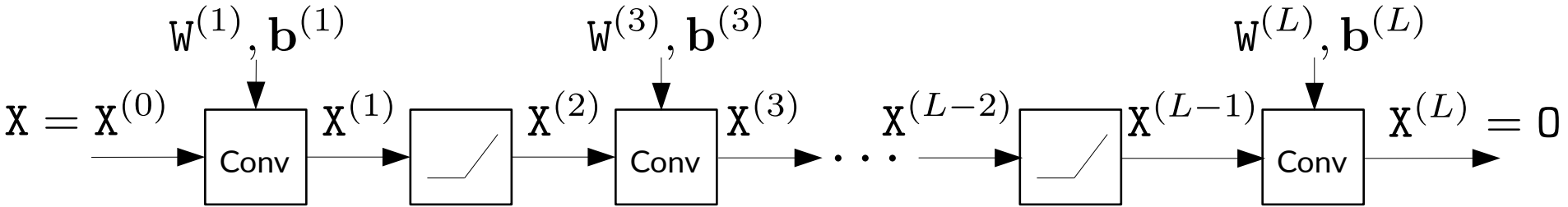


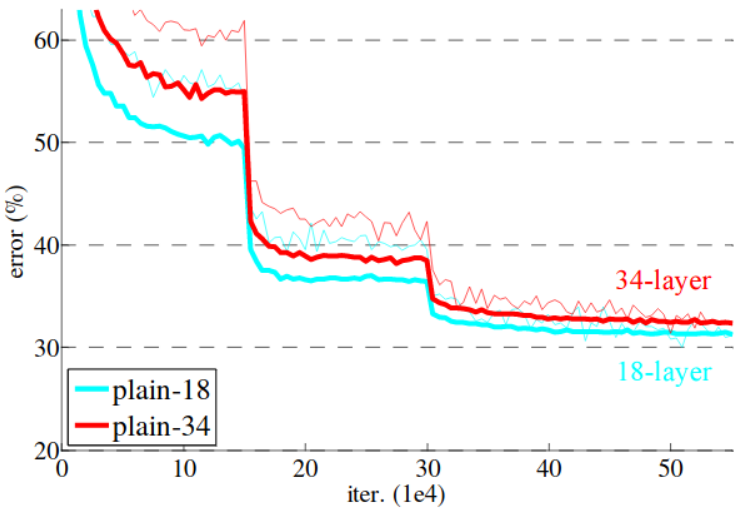


He, Kaiming, et al. "Deep residual learning for image recognition." Proceedings of the IEEE conference on computer vision and pattern recognition. 2016.

I)

Limites du CNN « classique »





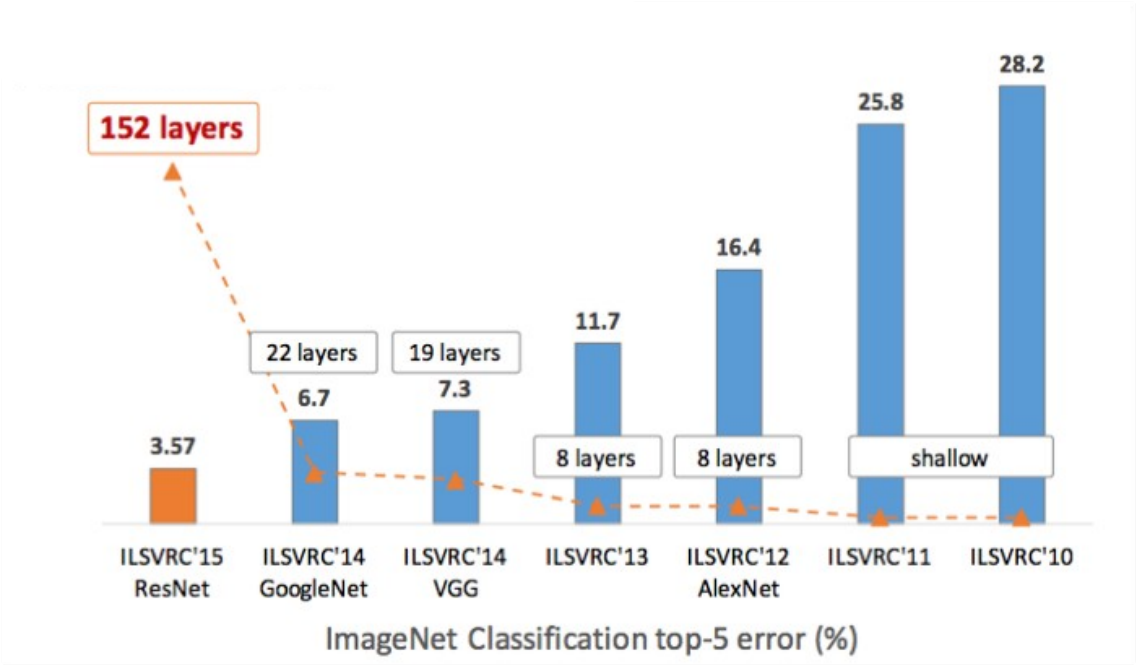
Ingrédient limitant les performances

Architecture du CNN

He, Kaiming, et al. "Deep residual learning for image recognition." Proceedings of the IEEE conference on computer vision and pattern recognition. 2016.

I)

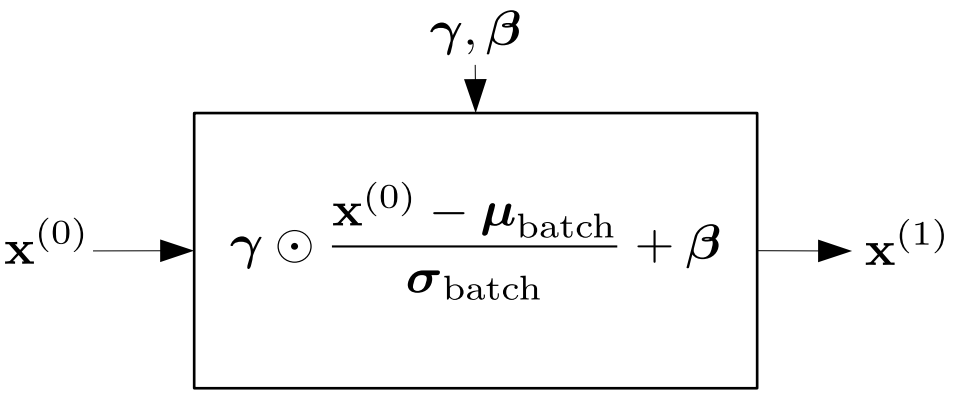
Limites du CNN « classique »



Source : https://medium.com/@Lidinwise/the-revolution-of-depth-facf174924f5

I)

Couche de “Batch Normalization”



Input: Values of x over a mini-batch: $\mathcal{B} = \{x_{1\ldots m}\}$;
Parameters to be learned: γ, β
Output: $\{y_i = \text{BN}_{\gamma,\beta}(x_i)\}$

$$\mu_{\mathcal{B}} \leftarrow \frac{1}{m}\sum_{i=1}^{m} x_i \qquad \text{// mini-batch mean}$$

$$\sigma^2_{\mathcal{B}} \leftarrow \frac{1}{m}\sum_{i=1}^{m}(x_i - \mu_{\mathcal{B}})^2 \qquad \text{// mini-batch variance}$$

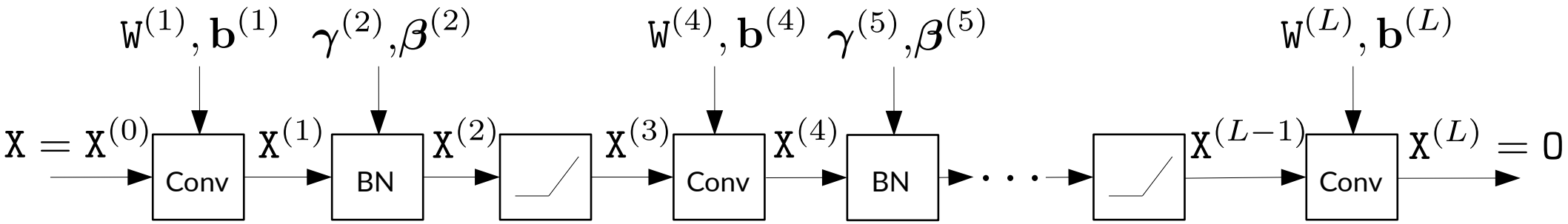
$$\widehat{x}_i \leftarrow \frac{x_i - \mu_{\mathcal{B}}}{\sqrt{\sigma^2_{\mathcal{B}} + \epsilon}} \qquad \text{// normalize}$$

$$y_i \leftarrow \gamma\widehat{x}_i + \beta \equiv \text{BN}_{\gamma,\beta}(x_i) \qquad \text{// scale and shift}$$

Ioffe, S., & Szegedy, C. (2015). Batch normalization: Accelerating deep network training by reducing internal covariate shift. arXiv preprint arXiv:1502.03167.

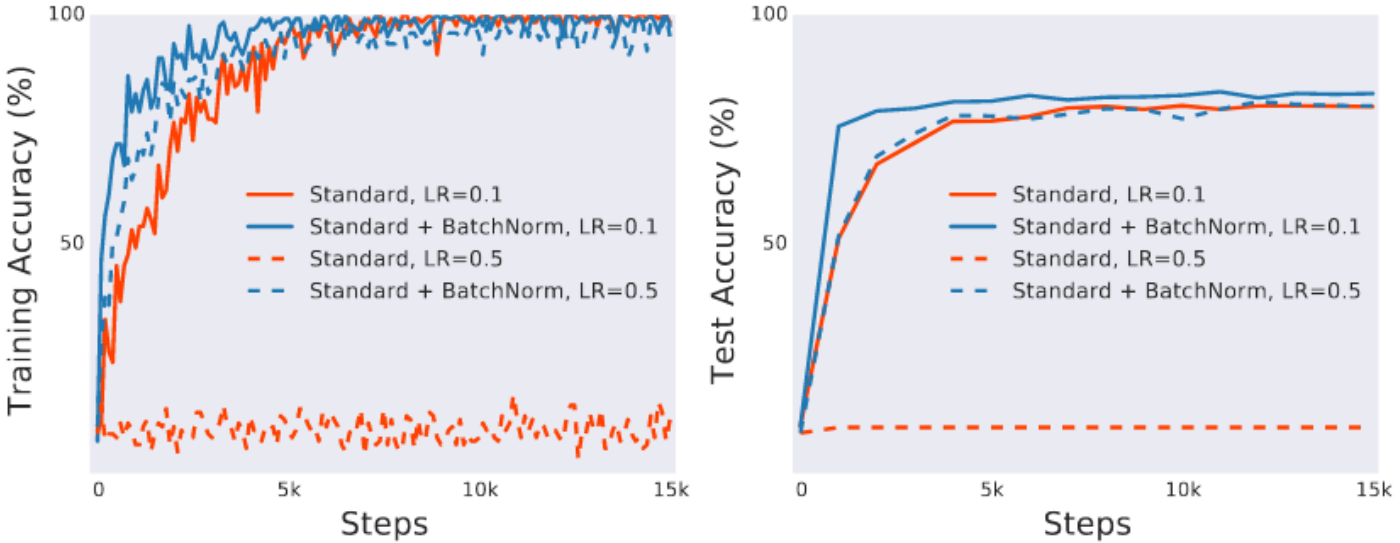
I)

CNN + BN


$\mathbb{W}^{(1)}, \mathbf{b}^{(1)}$
$\boldsymbol{\gamma}^{(2)}, \boldsymbol{\beta}^{(2)}$
$\mathbb{W}^{(4)}, \mathbf{b}^{(4)}$
$\boldsymbol{\gamma}^{(5)}, \boldsymbol{\beta}^{(5)}$
$\mathbb{W}^{(L)}, \mathbf{b}^{(L)}$
$\mathbb{X} = \mathbb{X}^{(0)}$
Conv
$\mathbb{X}^{(1)}$
BN
$\mathbb{X}^{(2)}$
$\mathbb{X}^{(3)}$
Conv
$\mathbb{X}^{(4)}$
BN
$\mathbb{X}^{(L-1)}$
Conv
$\mathbb{X}^{(L)} = \mathbb{O}$

I)

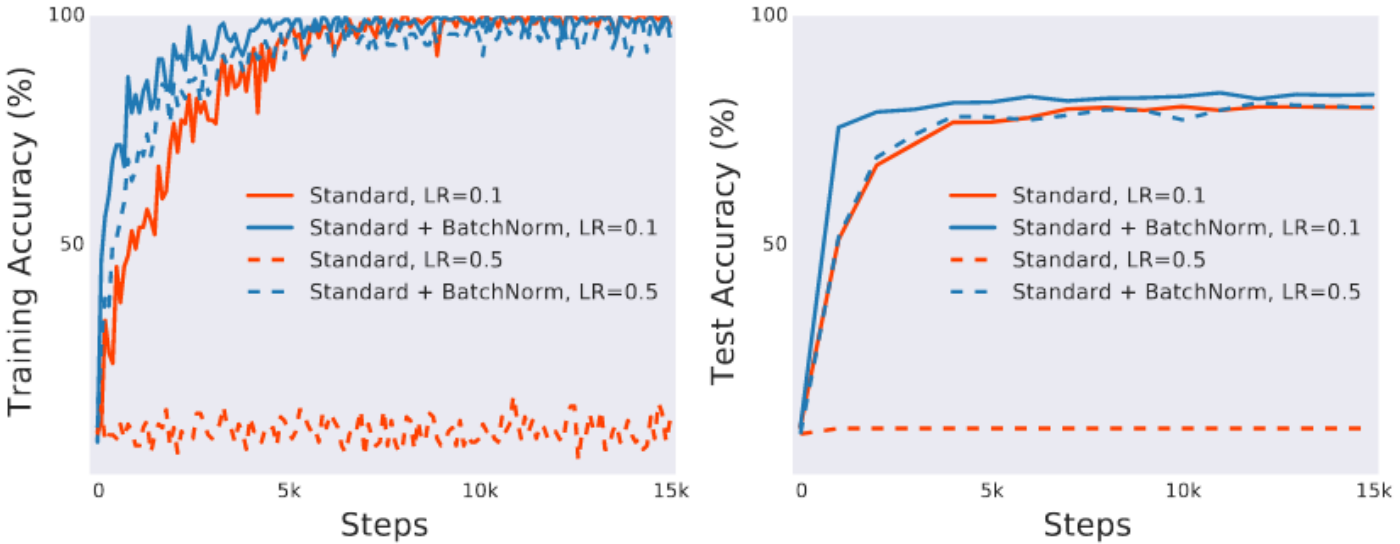
CNN + BN (suite)



Santurkar, Shibani, et al. "How does batch normalization help optimization?." Advances in neural information processing systems. 2018.

I)

CNN + BN (suite)



Rend le problème d'optimisation plus « lisse » :

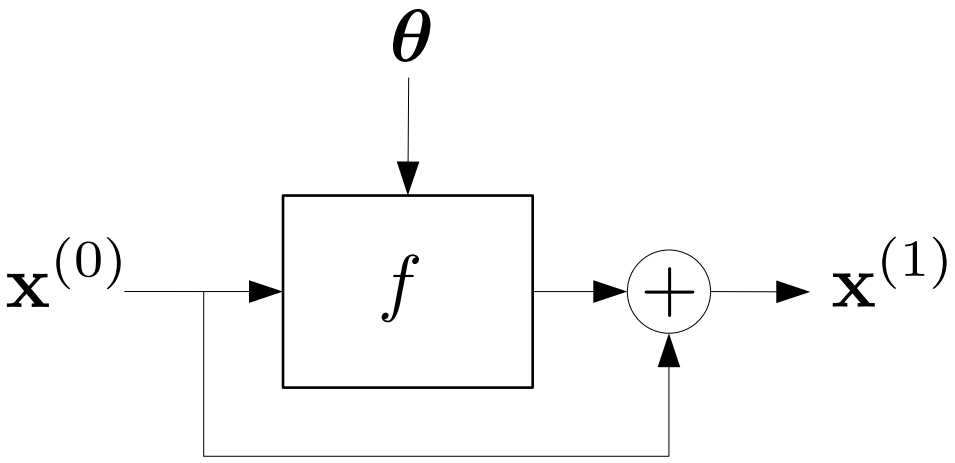
→ Initialisation des paramètres moins critique

→ Possibilité d'utilisation d'un plus grand pas d'apprentissage → accélération de l'entraînement

Santurkar, Shibani, et al. "How does batch normalization help optimization?." Advances in neural information processing systems. 2018.

I)

Connexion résiduelle

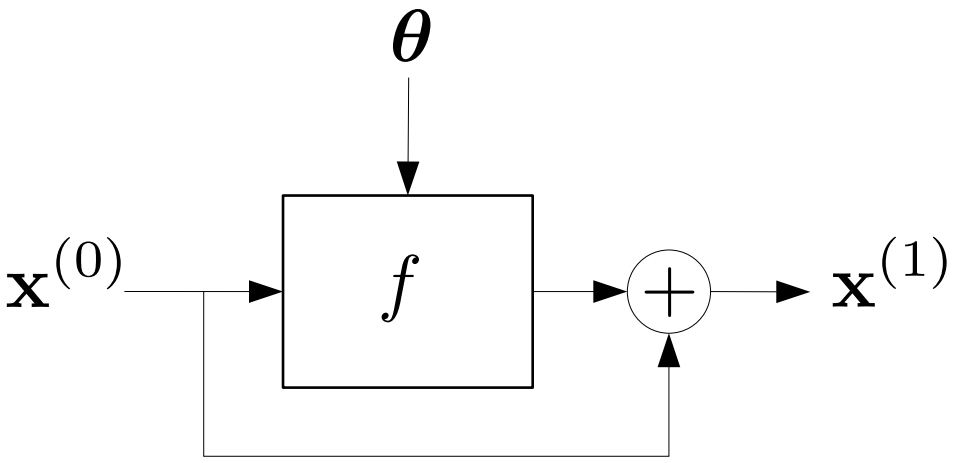


$$\mathbf{x}^{(1)} = \mathbf{x}^{(0)} + f(\mathbf{x}^{(0)}; \boldsymbol{\theta})$$

He, Kaiming, et al. "Deep residual learning for image recognition." Proceedings of the IEEE conference on computer vision and pattern recognition. 2016.

I)

Connexion résiduelle



$$\mathbf{x}^{(1)} = \mathbf{x}^{(0)} + f(\mathbf{x}^{(0)}; \boldsymbol{\theta})$$

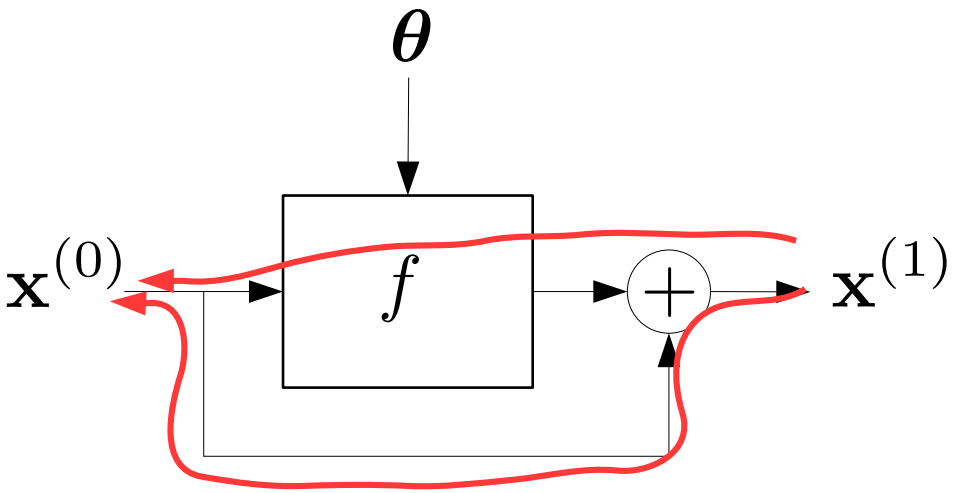
Rend la fonction plus « linéaire » :

→ Réduit sa « capacité » → augmentation du nombre de couches pour un même résultat

He, Kaiming, et al. "Deep residual learning for image recognition." Proceedings of the IEEE conference on computer vision and pattern recognition. 2016.

I)

Connexion résiduelle (suite)



$$\mathbf{x}^{(1)} = \mathbf{x}^{(0)} + f(\mathbf{x}^{(0)}; \boldsymbol{\theta})$$

$$\frac{\partial \mathbf{x}^{(1)}}{\partial \mathbf{x}^{(0)}} = \mathrm{I} + \frac{\partial f(\mathbf{x}^{(0)}; \boldsymbol{\theta})}{\partial \mathbf{x}^{(0)}}$$

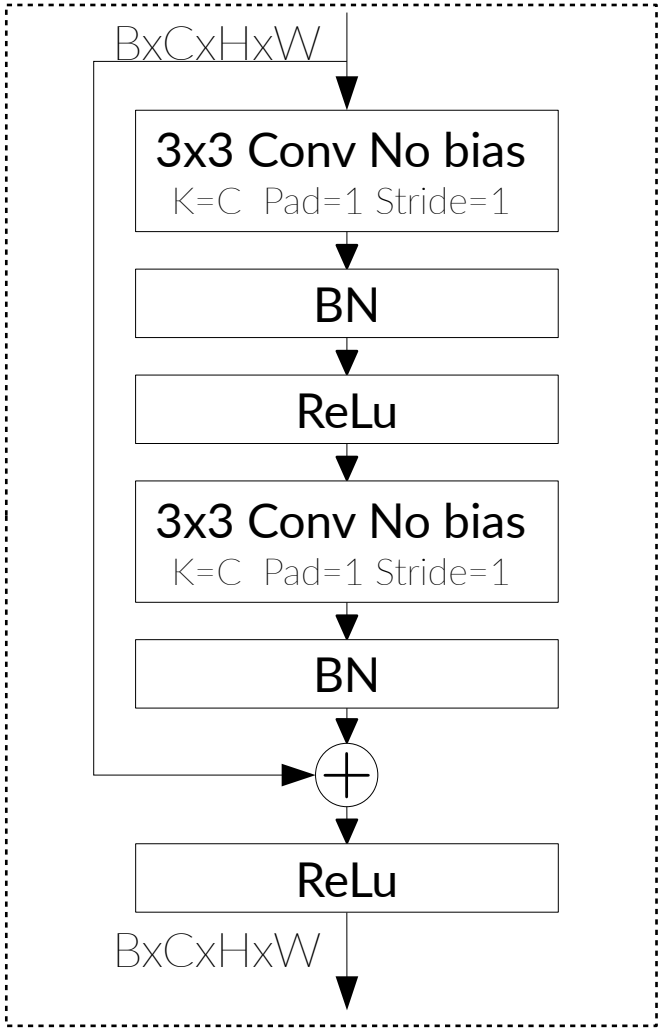
Rend la fonction plus « linéaire » :

→ Réduit sa « capacité » → augmentation du nombre de couches pour un même résultat

→ Facilite la propagation du gradient → plus de couches conduit à de meilleurs résultats (en théorie)

He, Kaiming, et al. "Deep residual learning for image recognition." Proceedings of the IEEE conference on computer vision and pattern recognition. 2016.

I)



ResBlock A

Transforme le tenseur d'entrée

- en préservant la résolution
- et en préservant le nombre de canaux

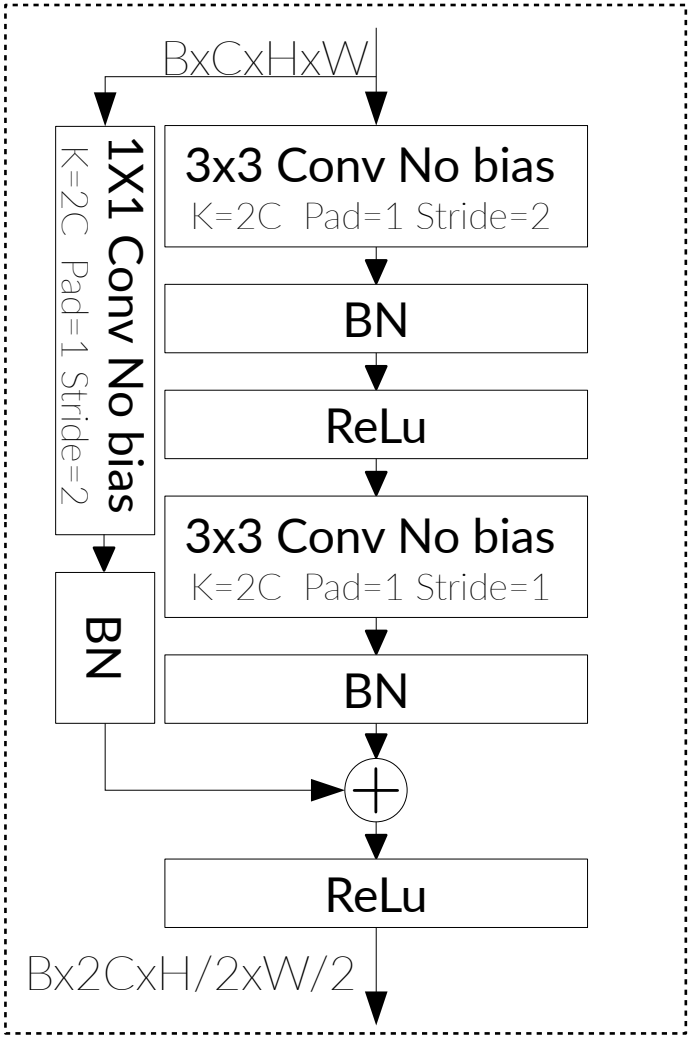
He, Kaiming, et al. "Deep residual learning for image recognition." Proceedings of the IEEE conference on computer vision and pattern recognition. 2016.

I)

ResBlock B

Transforme le tenseur d'entrée

- en divisant la résolution par 2
- et en augmentant le nombre de canaux par 2



He, Kaiming, et al. "Deep residual learning for image recognition." Proceedings of the IEEE conference on computer vision and pattern recognition. 2016.

I)

ResNet 18

$$\arg\max(\mathbf{o}) = 748 \doteq \text{"raie"}$$

Passage en grande dimension

3x224x224

64x112x112

64x112x112

64x112x112

64x56x56

64x56x56

64x56x56

128x28x28

7x7 Conv No bias
K=64 Pad=3 Stride=2

BN

ReLu

3x3 MaxPool
Pad=1 Stride=2

ResBlock A

ResBlock A

ResBlock B

ResBlock A

ResBlock B

ResBlock B

ResBlock A

7x7 Average Pool

1000x512 FC

$\mathbf{o}$

128x28x28

256x14x14

256x14x14

512x7x7

512x7x7

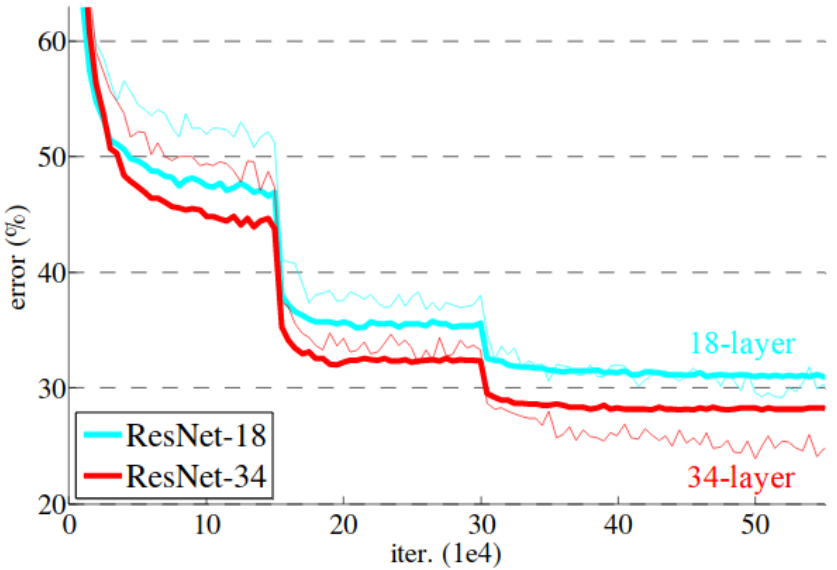
512x1

1000x1

He, Kaiming, et al. "Deep residual learning for image recognition." Proceedings of the IEEE conference on computer vision and pattern recognition. 2016.

I)

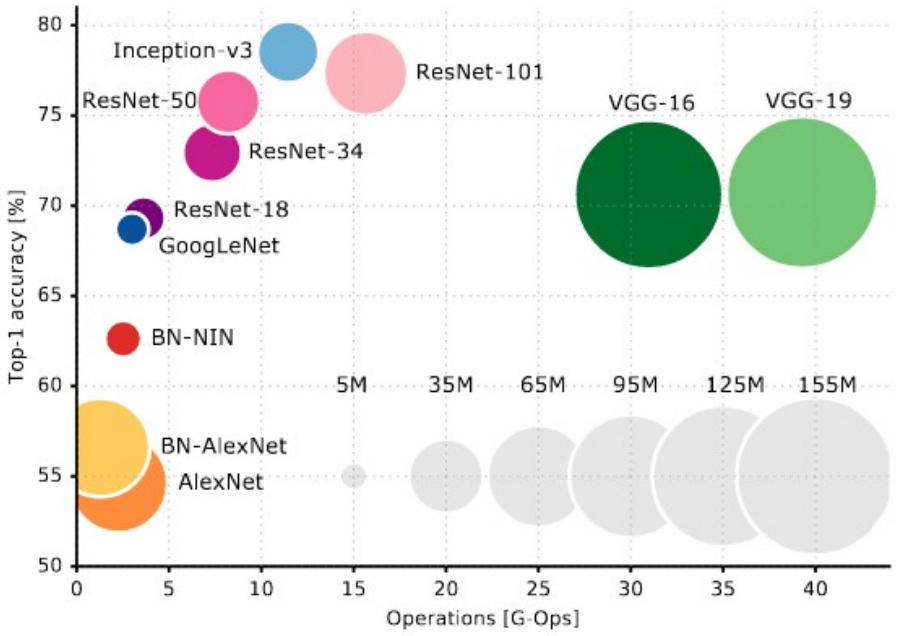
ResNet 18 < ResNet 34



He, Kaiming, et al. "Deep residual learning for image recognition." Proceedings of the IEEE conference on computer vision and pattern recognition. 2016.

I)

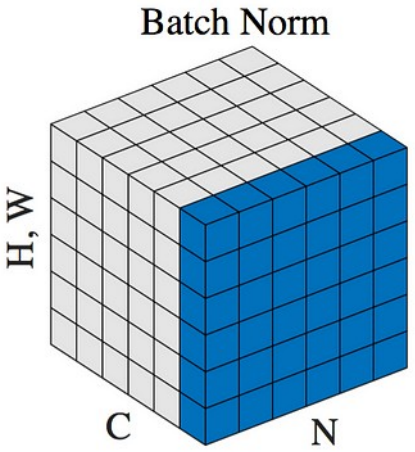
Précision vs Nombre de paramètres vs Nombre d'opérations



Canziani, A., Paszke, A., & Culurciello, E. (2016). An analysis of deep neural network models for practical applications. arXiv preprint arXiv:1605.07678.

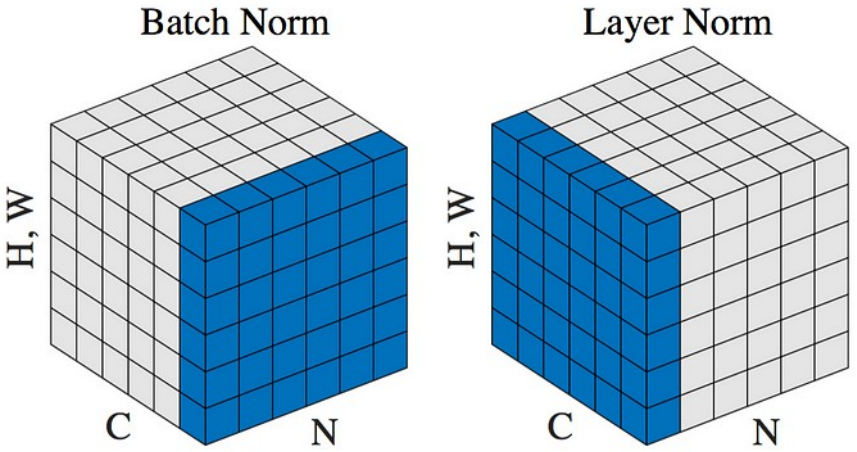
I)

Différentes couches de normalisation



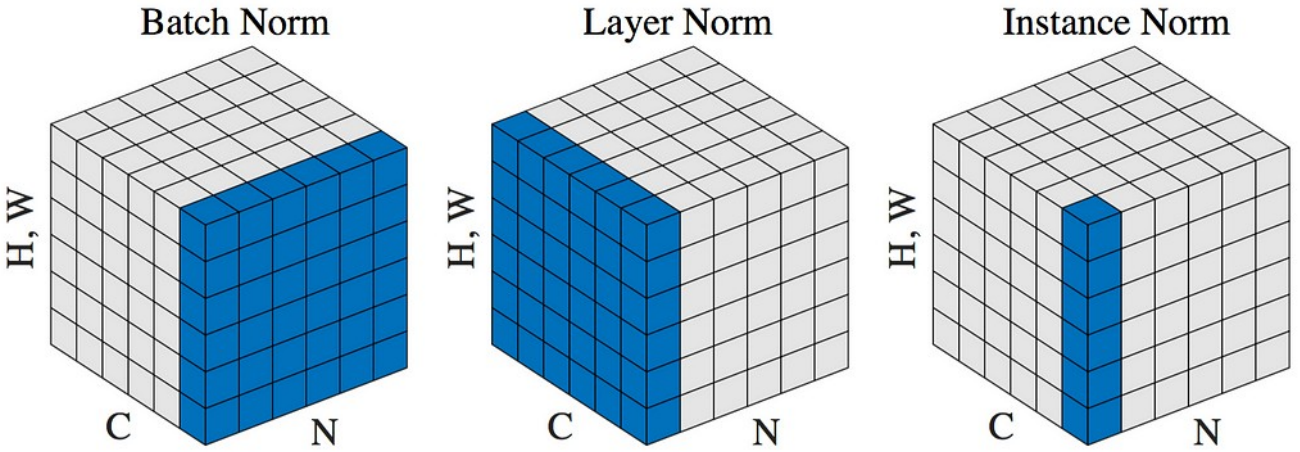
I)

Différentes couches de normalisation



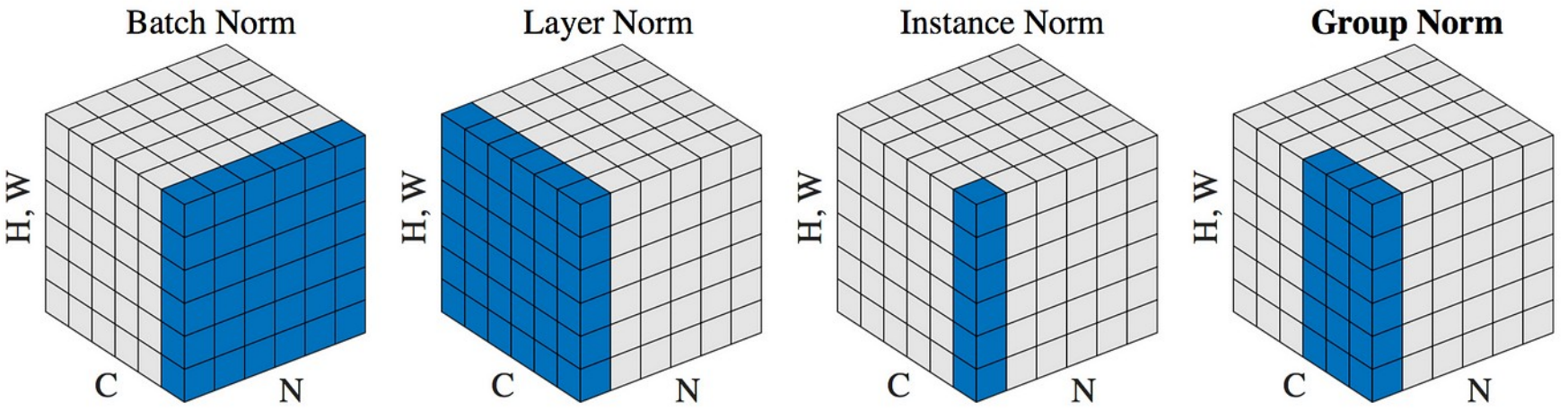
I)

Différentes couches de normalisation



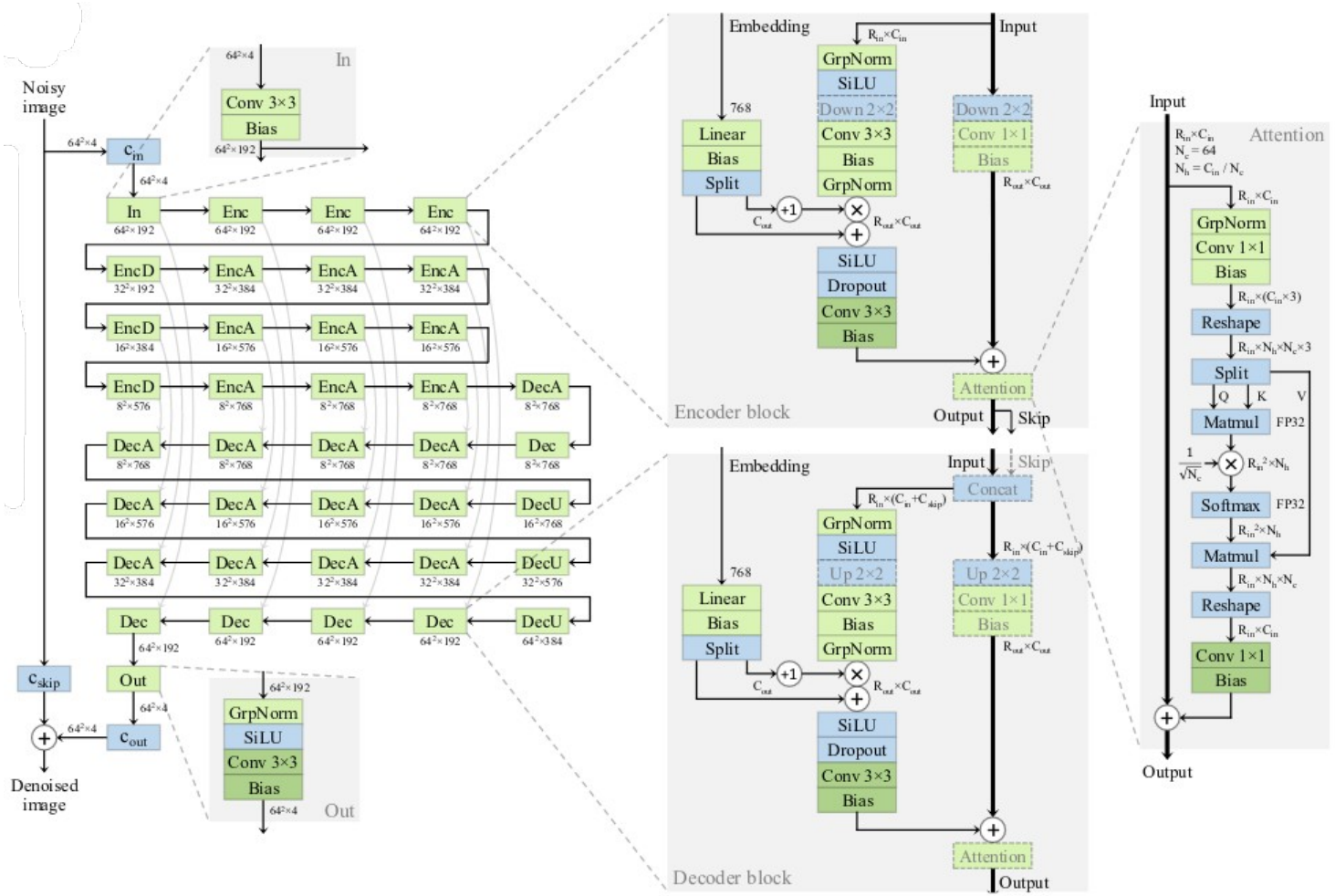
I)

Différentes couches de normalisation



I)

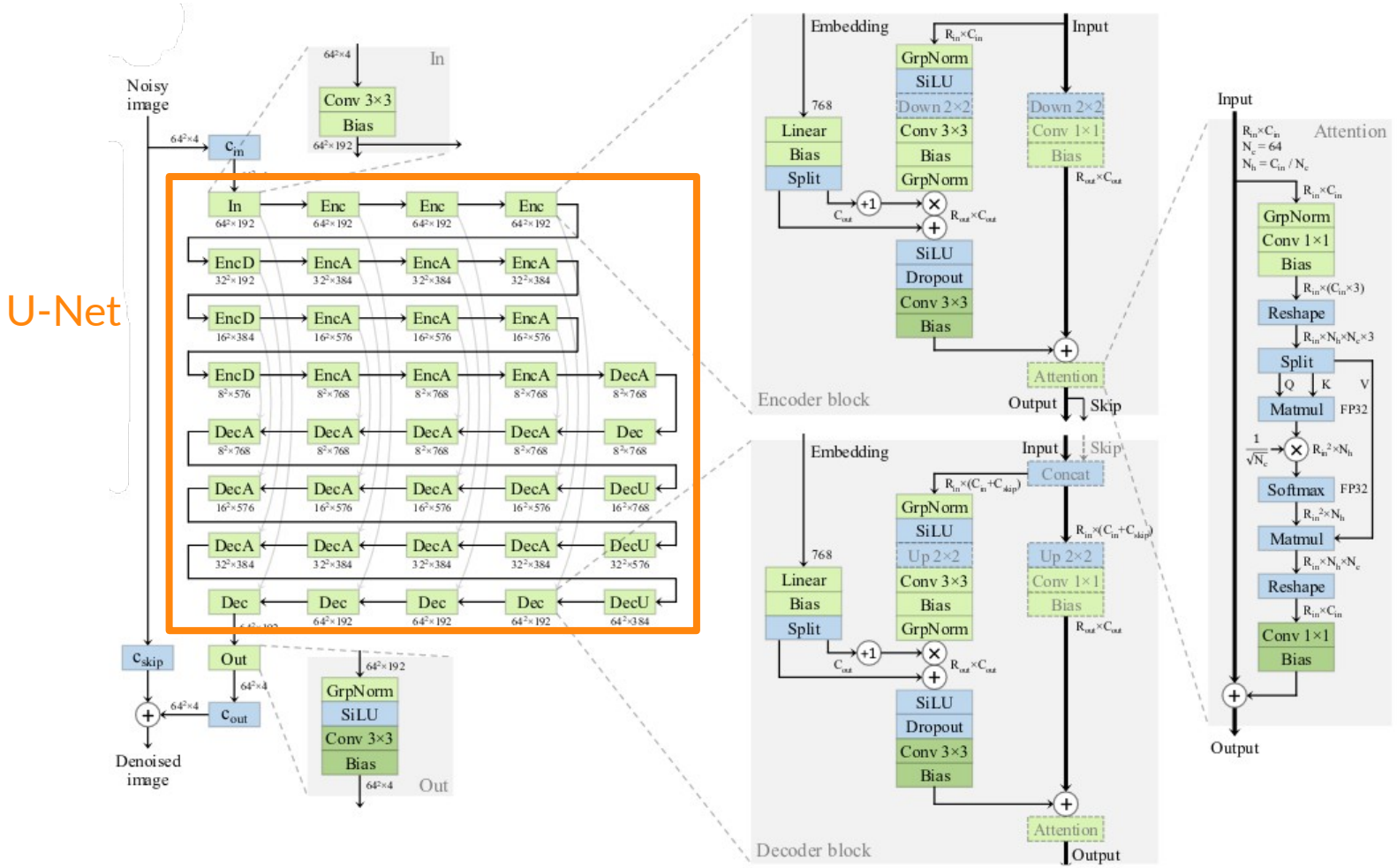
En 2024 : U-Net + ResBlock toujours là !



Karras et al. (2024). Analyzing and Improving the Training Dynamics of Diffusion Models arXiv preprint arXiv:2312.02696.

I)

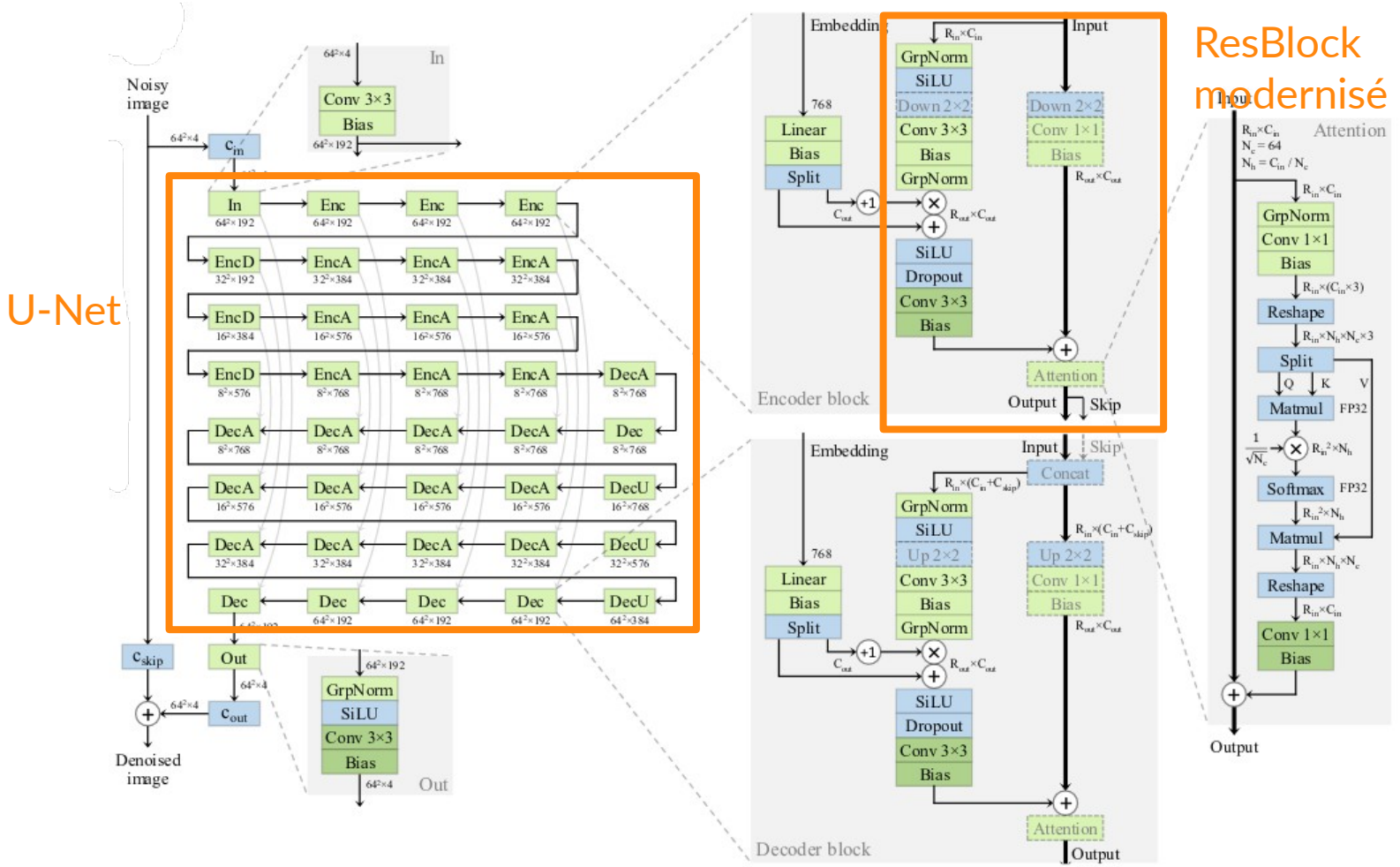
En 2024 : U-Net + ResBlock toujours là !



Karras et al. (2024). Analyzing and Improving the Training Dynamics of Diffusion Models arXiv preprint arXiv:2312.02696.

I)

En 2024 : U-Net + ResBlock toujours là !



Karras et al. (2024). Analyzing and Improving the Training Dynamics of Diffusion Models arXiv preprint arXiv:2312.02696.

II) Modèles de fondation

II)

Comment faire avec une petite base de données étiquetées ?

Exemple : détection du frelon asiatique

Présence (1042 images)

Absence (1844 images)

Ingrédient limitant les performances

Taille de la base de données étiquetées

II)

Comment faire avec une petite base de données étiquetées ?

Exemple : détection du frelon asiatique

Présence (1042 images)

Absence (1844 images)

Ingrédient limitant les performances

↓

Taille de la base de données étiquetées

Solution

1) Récupérer un modèle de fondation

2) Spécialiser ce modèle de fondation sur sa petite base de données étiquetées

II)

Modèle de fondation « historique »

Historiquement (~ à partir de 2012), un modèle de fondation (le terme « foundation model » n'est introduit qu'en 2021) est :

II)

Modèle de fondation « historique »

Historiquement (~ à partir de 2012), un modèle de fondation (le terme « foundation model » n'est introduit qu'en 2021) est :

- un CNN (e.g ResNet)

II)

Modèle de fondation « historique »

Historiquement (~ à partir de 2012), un modèle de fondation (le terme « foundation model » n'est introduit qu'en 2021) est :

- un CNN (e.g ResNet)
- qui est **pré-entraîné =** entraîné sur ImageNet1k

Rappel : ImageNet1k = 1.2M d'images étiquetées sur 1000 classes qui représentent une grande diversité d'images **issues de notre monde**

II) Exemple de spécialisation (« fine-tuning ») d'un ResNet 18 pré-entraîné sur ImageNet1k

3x224x224

7x7 Conv No bias
K=64 Pad=3 Stride=2

64x112x112

BN

64x112x112

ReLu

64x112x112

3x3 MaxPool
Pad=1 Stride=2

64x56x56

ResBlock A

64x56x56

ResBlock A

64x56x56

ResBlock B

128x28x28

ResBlock A

128x28x28

ResBlock B

256x14x14

ResBlock A

256x14x14

ResBlock B

512x7x7

ResBlock A

512x7x7

7x7 Average Pool

512x1

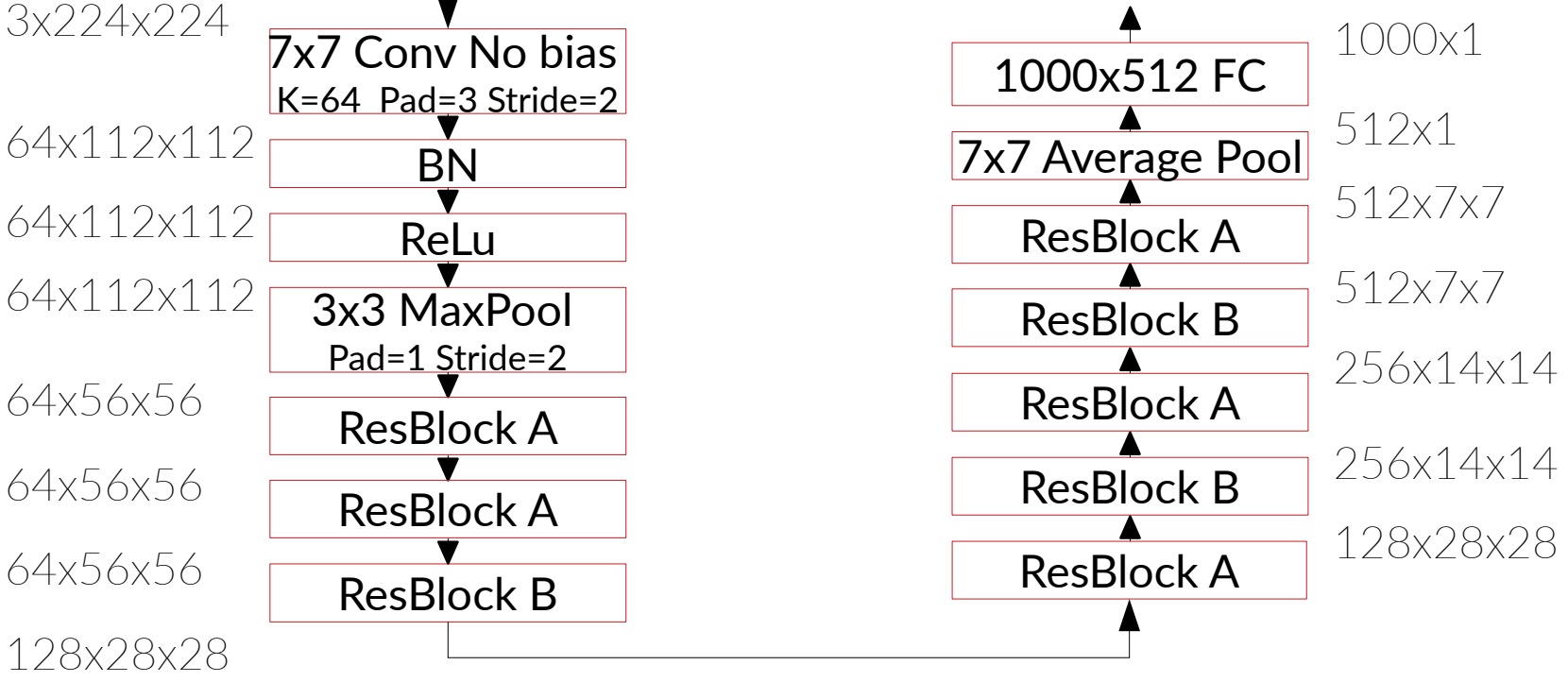
1000x512 FC

1000x1

II) Exemple de spécialisation (« fine-tuning ») d'un ResNet 18 pré-entraîné sur ImageNet1k

Image « normalisée »

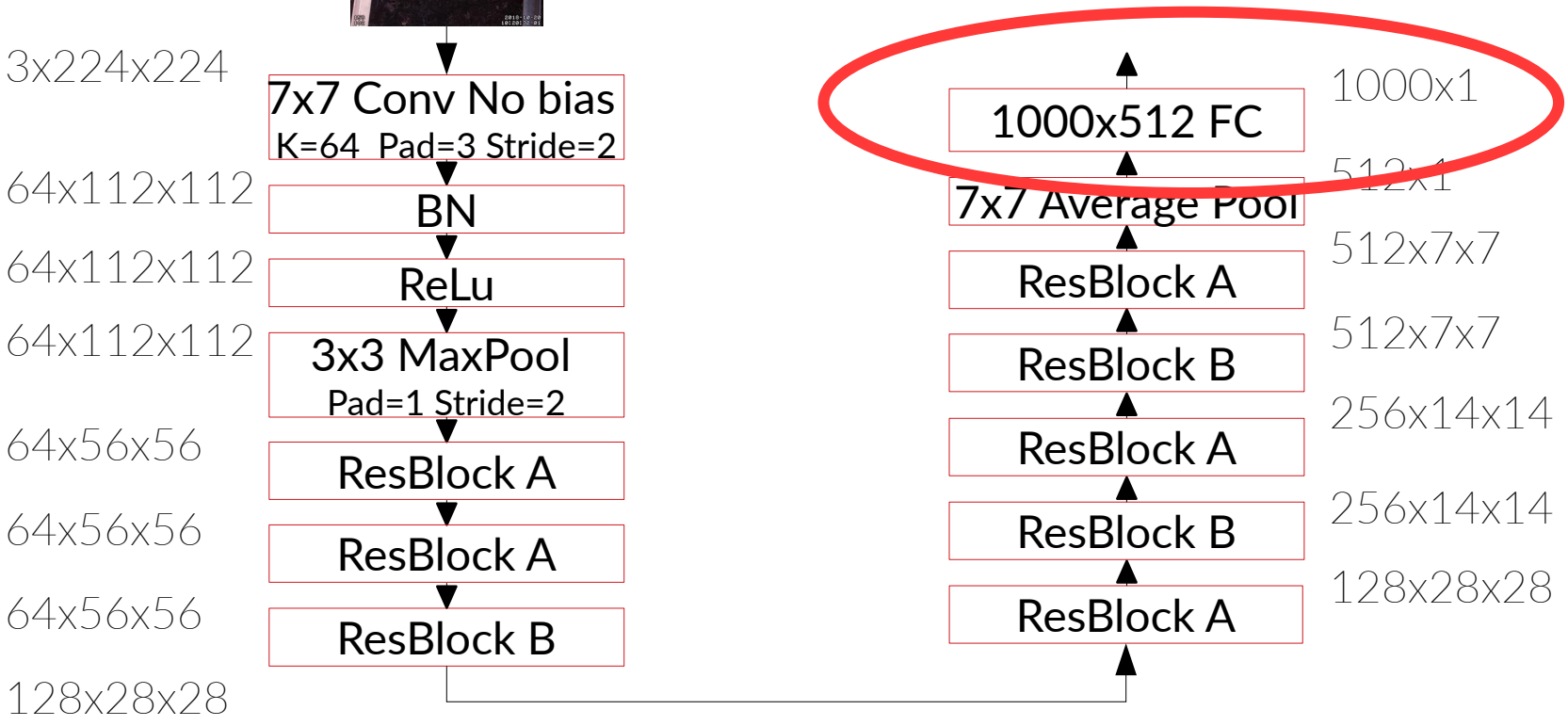
Exemple : réseau pré-entraîné sur images [-1, 1]
→ normalisation de mes images dans [-1, 1]



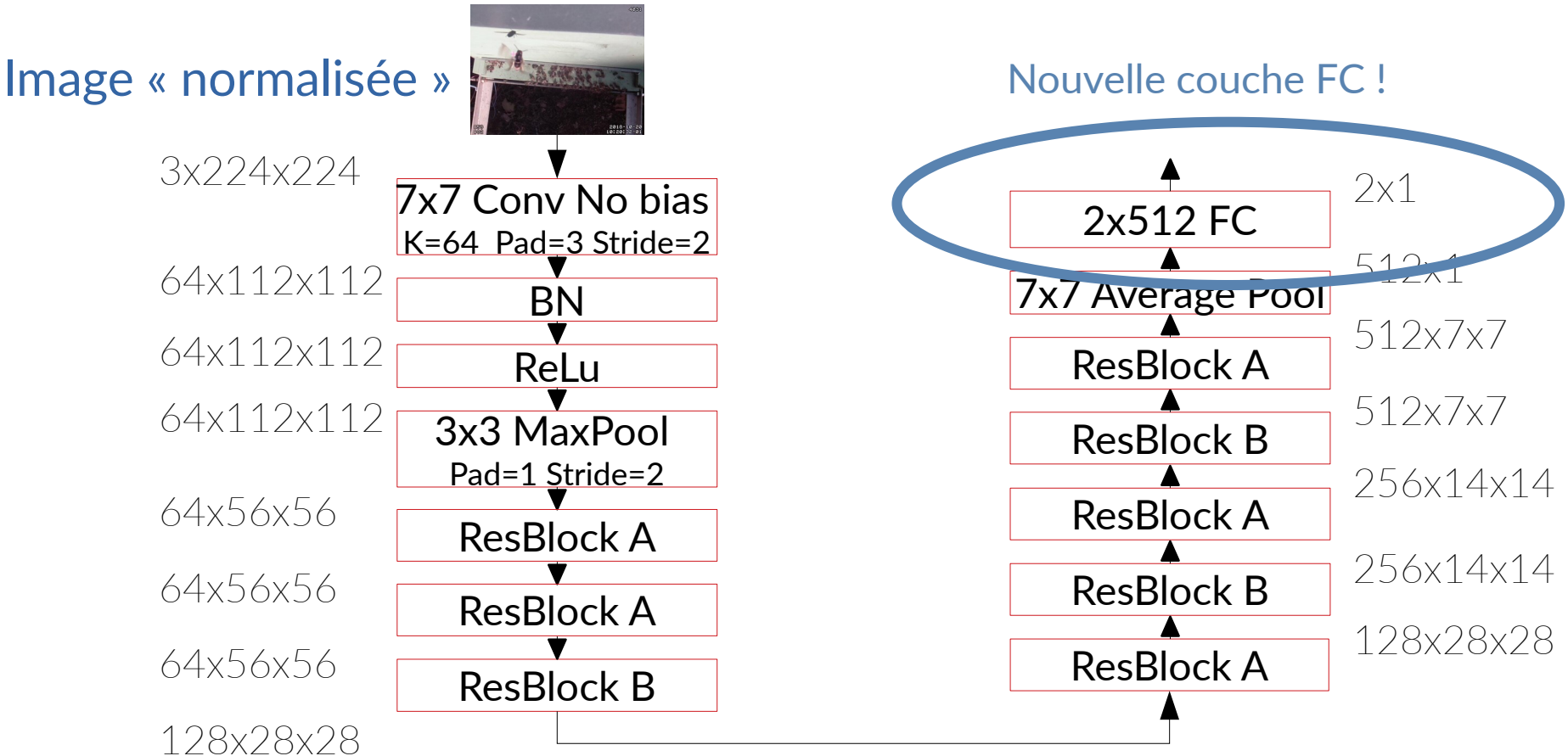
II) Exemple de spécialisation (« fine-tuning ») d'un ResNet 18 pré-entraîné sur ImageNet1k

Image « normalisée »

Problème : on veut prédire 2 scores (présence de frélon, absence de frelon), pas 1000 scores...



II) Exemple de spécialisation (« fine-tuning ») d'un ResNet 18 pré-entraîné sur ImageNet1k



II) Exemple de spécialisation (« fine-tuning ») d'un ResNet 18 pré-entraîné sur ImageNet1k

Image « normalisée »

3x224x224

7x7 Conv No bias
K=64 Pad=3 Stride=2

64x112x112

BN

64x112x112

ReLu

64x112x112

3x3 MaxPool
Pad=1 Stride=2

64x56x56

ResBlock A

64x56x56

ResBlock A

64x56x56

ResBlock B

128x28x28

$$\arg\max(\mathbf{o}) = 2 \doteq \text{"Présence"}$$

2x1

2x512 FC

512x1

7x7 Average Pool

512x7x7

ResBlock A

512x7x7

ResBlock B

256x14x14

ResBlock A

256x14x14

ResBlock B

128x28x28

ResBlock A

Paramètres initialisés par les valeurs d'un ResNet18 entraîné sur ImageNet

Initialisation classique (aléatoire)

II)

Modèles de fondation « modernes »

Pré-entraînement sur une **très** grande base de données.

Exemple : ~2012 ImageNet 1k = 1.2x10^6 images étiquetées
~2022 LAION-5B = 5x10^9 images étiquetées

II)

Modèles de fondation « modernes »

Pré-entraînement sur une **très** grande base de données.

Exemple : ~2012 ImageNet 1k = 1.2x10^6 images étiquetées
~2022 LAION-5B = 5x10^9 images étiquetées

Rendu possible grâce à :

- l'augmentation continue de la puissance des cartes graphiques
- l'apparition des couches d'attention (« Transformer »)

II)

Modèles de fondation « modernes »

Pré-entraînement sur une **très** grande base de données.

Exemple : ~2012 ImageNet 1k = 1.2x10^6 images étiquetées
~2022 LAION-5B = 5x10^9 images étiquetées

Rendu possible grâce à :

- l'augmentation continue de la puissance des cartes graphiques
- l'apparition des couches d'attention (« Transformer »)

Apparition de modèles de fondation dans un grand nombre de domaines :

- Imagerie médicale
- Imagerie astronomique
- Etc.

II)

Modèles de fondation « modernes »

Pré-entraînement sur une **très** grande base de données.

Exemple : ~2012 ImageNet 1k = 1.2x10^6 images étiquetées
~2022 LAION-5B = 5x10^9 images étiquetées

Rendu possible grâce à :

- l'augmentation continue de la puissance des cartes graphiques
- l'apparition des couches d'attention (« Transformer »)

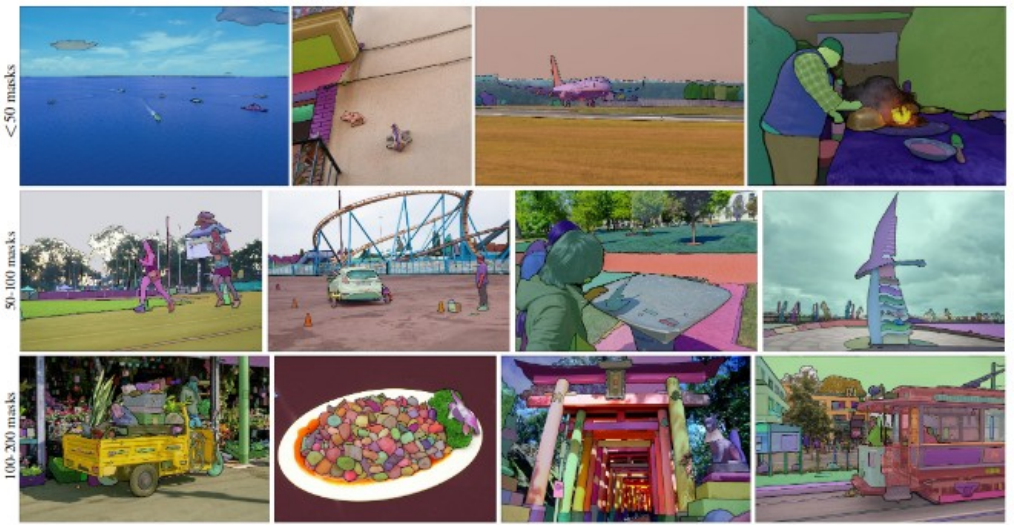
Apparition de modèles de fondation dans un grand nombre de domaines :

- Imagerie médicale
- Imagerie astronomique
- Etc.
- Et bien-sûr les LLM (« Large Language Model ») avec GPT !

II) Exemple de modèle de fondation pour la segmentation d'images

-

« Segment Anything »



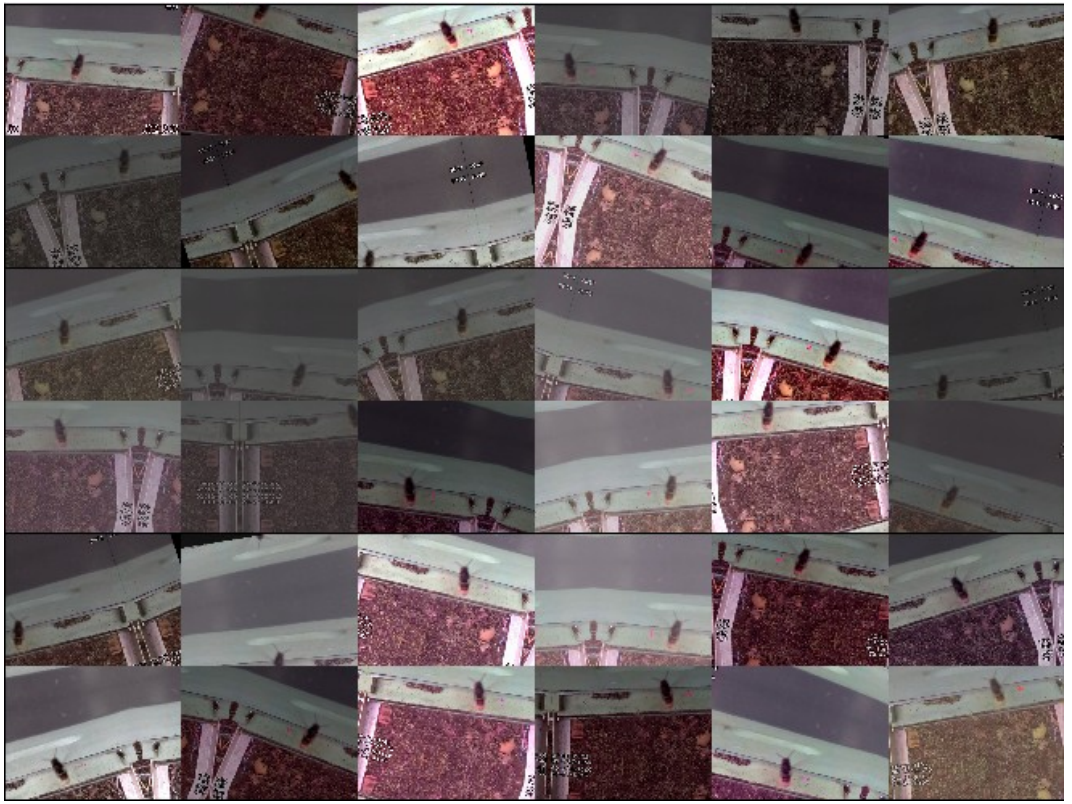
Entraînement sur « SA-1B » : une base de 11×10^6 images avec 1.1×10^9 masques de segmentation

Kirillov, Alexander, et al. "Segment Anything." CVPR 2023

Annexes

Augmentation de données

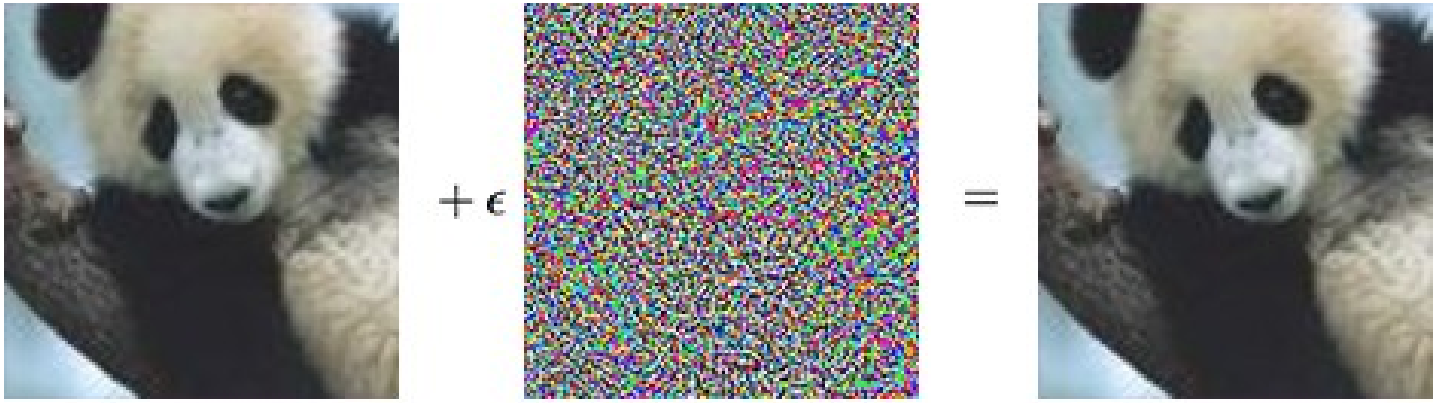
- Mirroir
- Transformation affine
- Perturbation couleur
- Effacement
- Bruit
- …



Cubuk et al. (2019). Autoaugment: Learning augmentation strategies from data. CVPR

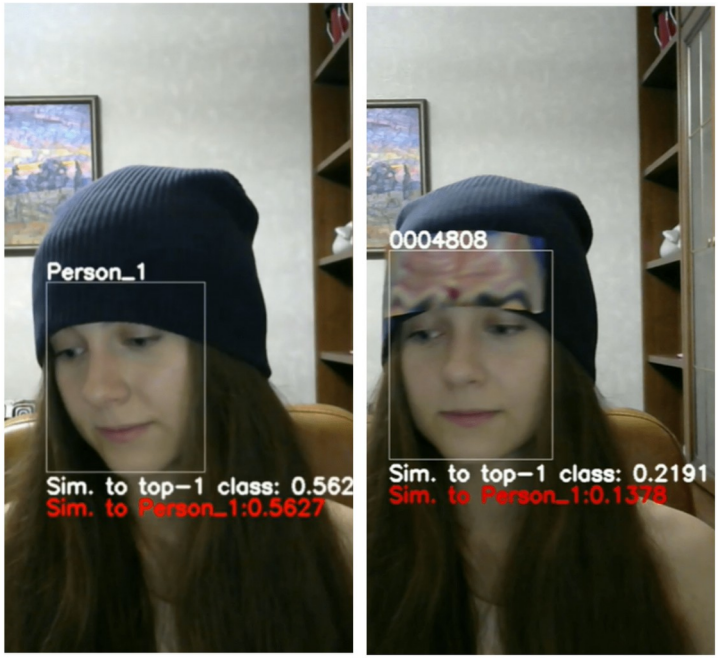
Cubuk et al. (2020). Randaugment: Practical automated data augmentation with a reduced search space. CVPR

“Adversarial examples”



Source : https://openai.com/blog/adversarial-example-research/

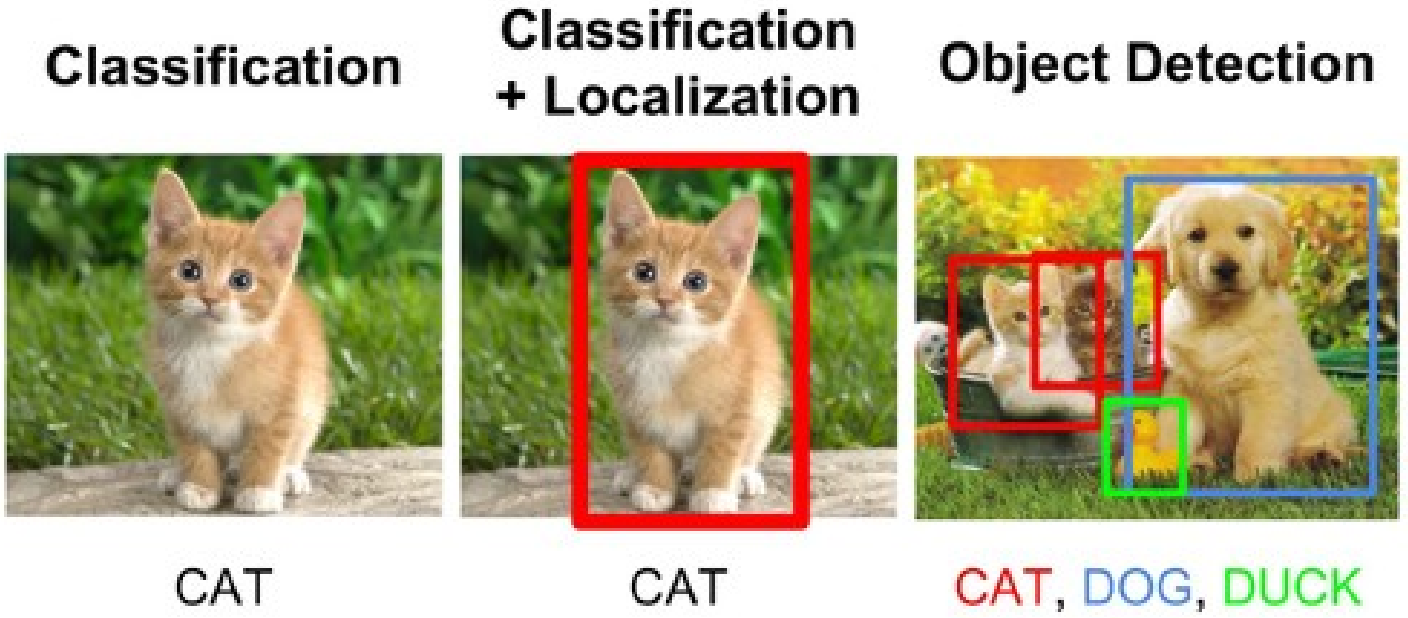
“Adversarial patches”



Sources : “ADVHAT: Real-world adversarial attack on ArcFace face ID system” (https://arxiv.org/pdf/1908.08705.pdf)
”Robust Physical-World Attacks on Deep Learning Visual Classification” (https://arxiv.org/pdf/1707.08945.pdf)

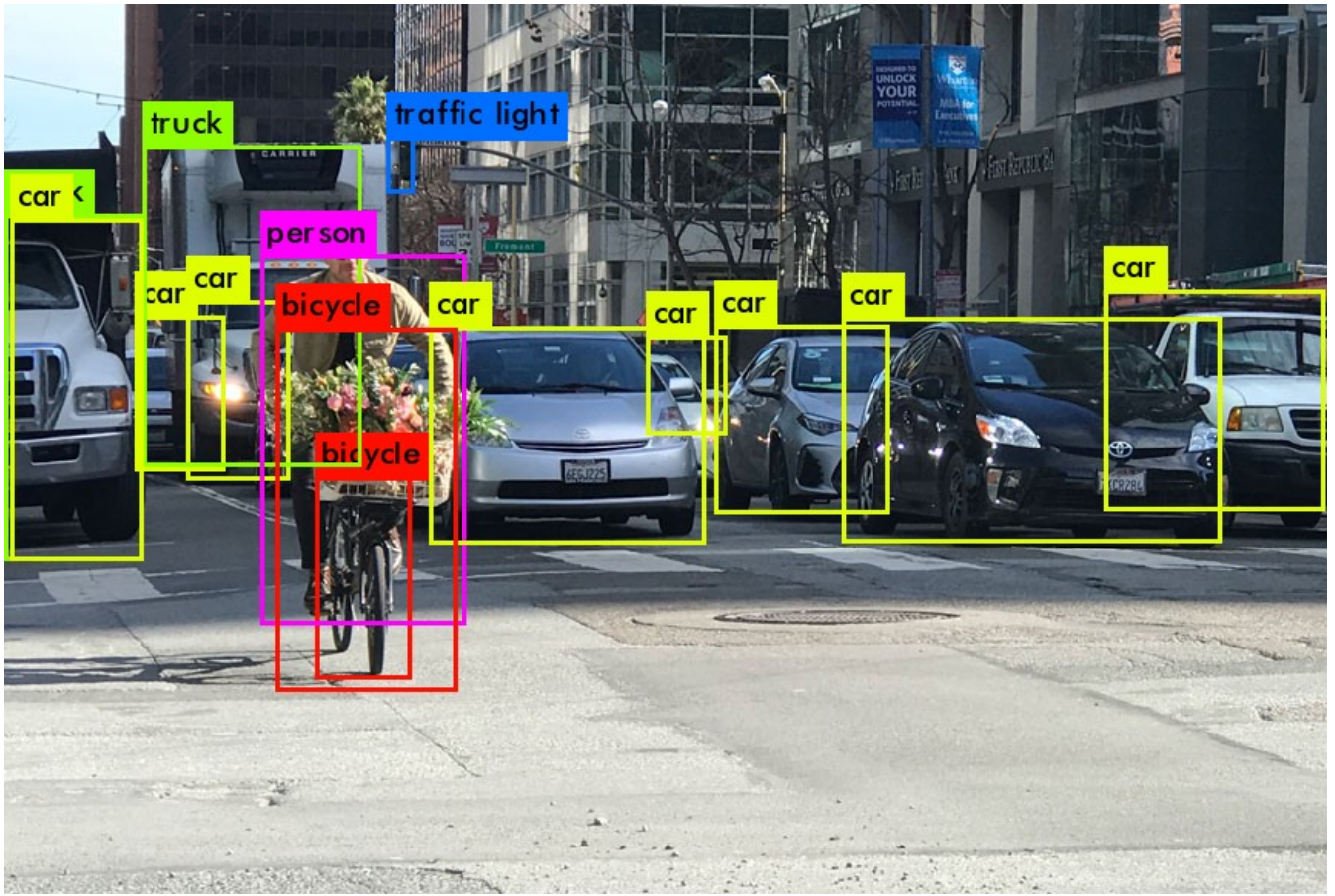
Annexe : Application à la détection d’objets dans une image

Détection d’objets



Source : https://medium.com/analytics-vidhya/yolov3-real-time-object-detection-54e69037b6d0

Exemple de détection d'objets



Source : https://towardsdatascience.com/yolo-you-only-look-once-17f9280a47b0

Formulation du problème

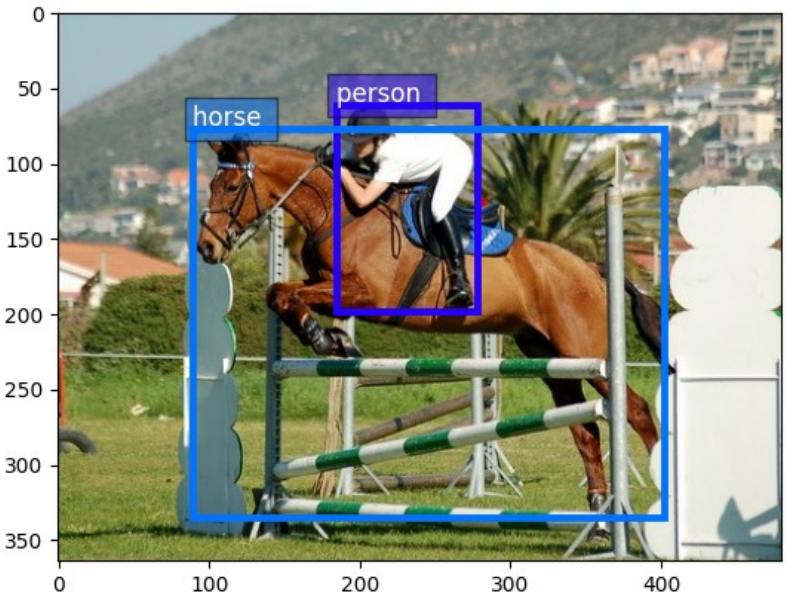
- Objectif
  - Prédire une boîte englobante autour de chaque objet,
  - Prédire la classe de l'objet,
  - En utilisant un réseau de neurones en apprentissage supervisé.

- Questions
  - Quelles données annotées disponibles ?
  - Quelle architecture ?
  - Quelle fonction de coût ?

Quelles données annotées disponibles ?

Microsoft COCO dataset (80 classes)

1: 'person',
2: 'bicycle',
3: 'car',
4: 'motorcycle',
5: 'airplane',
6: 'bs',
7: 'train',
8: 'trck',
9: 'boat',
10: 'traffic light',
11: 'fire hydrant',
12: 'stop sign',
13: 'parking meter',
14: 'bench',
15: 'bird',
16: 'cat',
17: 'dog',
18: 'horse',
19: 'sheep',
20: 'cow',
21: 'elephant',
22: 'bear',
23: 'zebra',
24: 'giraffe',
25: 'backpack',
26: 'mbrella',
31: 'skis',
32: 'snowboard',
33: 'sports ball',
34: 'kite',
35: 'baseball bat',
36: 'baseball glove',
37: 'skateboard',
38: 'srfboard',
39: 'tennis racket',
40: 'bottle',
41: 'wine glass',
42: 'cp',
43: 'fork',
44: 'knife',
45: 'spoon',
46: 'bowl',
47: 'banana',
48: 'apple',
49: 'sandwich',
50: 'orange',
51: 'broccoli',
52: 'carrot',
53: 'hot dog',
54: 'pizza',
55: 'dont',
56: 'cake',
61: 'dining table',
62: 'toilet',
63: 'tv',
64: 'laptop',
65: 'mose',
66: 'remote',
67: 'keyboard',
68: 'cell phone',
69: 'microwave',
70: 'oven',
71: 'toaster',
72: 'sink',
73: 'refrigerator',
74: 'book',
75: 'clock',
76: 'vase',
77: 'scissors',
78: 'teddy bear',
79: 'hair drier',
80: 'toothbrsh'



Pascal VOC dataset (20 classes)

Person:
- 1: person

Animal:
- 2: bird
- 3: cat
- 4: cow
- 5: dog
- 6: horse
- 7: sheep

Vehicle:
- 8: aeroplane
- 9: bicycle
- 10: boat
- 11: bus
- 12: car
- 13: motorbike
- 14: train

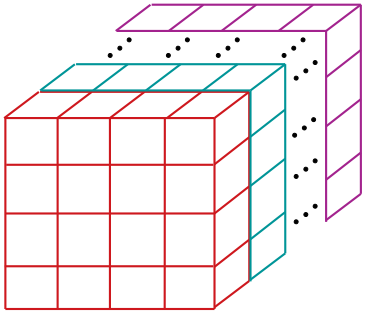
Indoor:
- 15: bottle
- 16: chair
- 17: dining table
- 18: potted plant

Comment prédire le nombre d'objets et pour chaque objet sa boîte et sa classe ?

Exemple de solution : CenterNet (Zhou et. al, Objects as points, 2019)

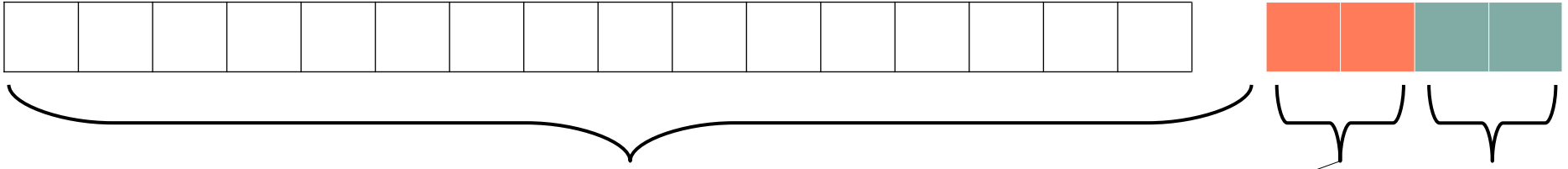
keypoint heatmap [C] local offset [2] object size [2]

CenterNet



3 x H x W

(C+2+2) x H/4 x W/4

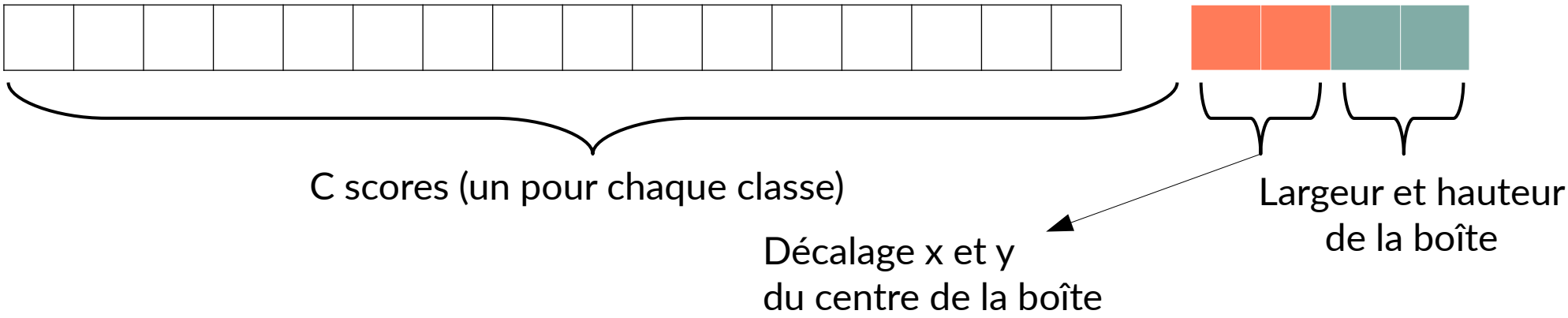


C scores (un pour chaque classe)

Décalage x et y
du centre de la boîte

Largeur et hauteur
de la boîte

CenterNet (suite)



Fonction de coût = somme de trois fonctions de coût

- Pour les scores : « pixelwise logistic regression » (i.e. sur chaque case grise)

- Pour le décalage : régression L1 (sur les cases oranges s'il y a une boîte sinon rien)

- Pour la largeur et la hauteur : régression L1 (sur les cases vertes s'il y a une boîte sinon rien)

CenterNet (suite)