UNIVERSITÉ LAVAL

GLO-4030/7030
APPRENTISSAGE PAR RÉSEAUX DE NEURONES PROFONDS

CNN (Partie II) :
Exemples d'architecture

Philippe Giguère

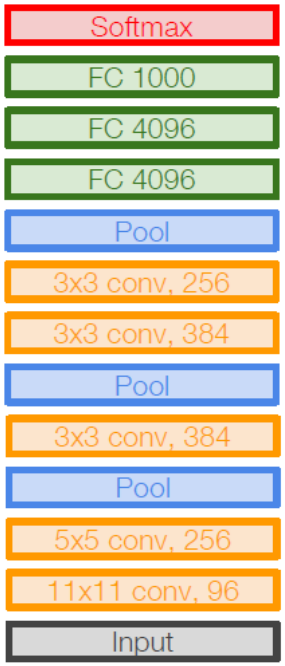
Admin

- Pas de laboratoire cette semaine
- Examen la semaine prochaine

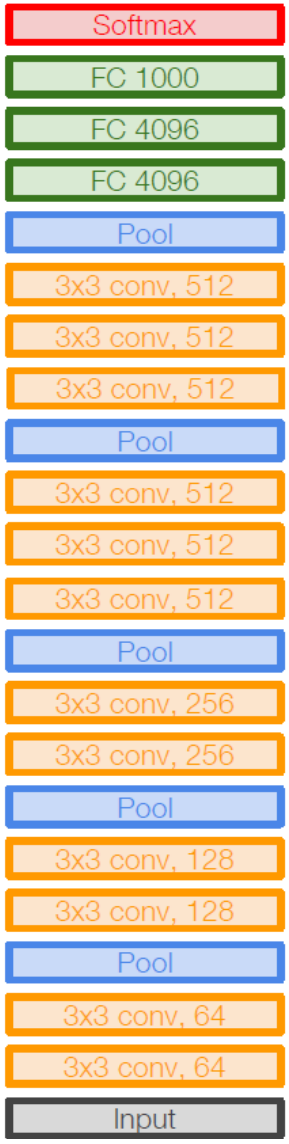
Résumé CNN I

- Plus grande profondeur
- Disparition graduelle des têtes fully-connected
  - remplacé par Global Average Pooling + 1 layer de fully-connected
- conv 3x3 est la taille dominante
- Parfois conv plus grande que 3x3 à la base
  - 5x5 ou 7x7

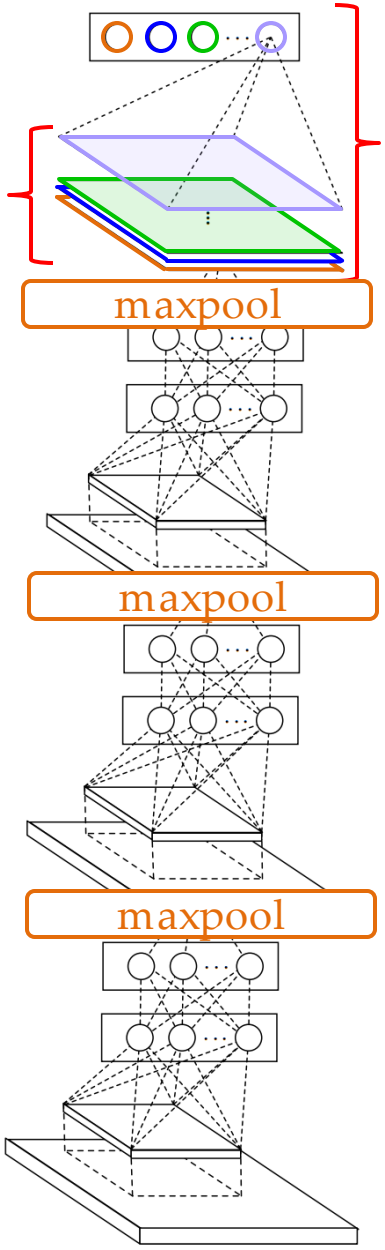
Résumé des architectures CNN I



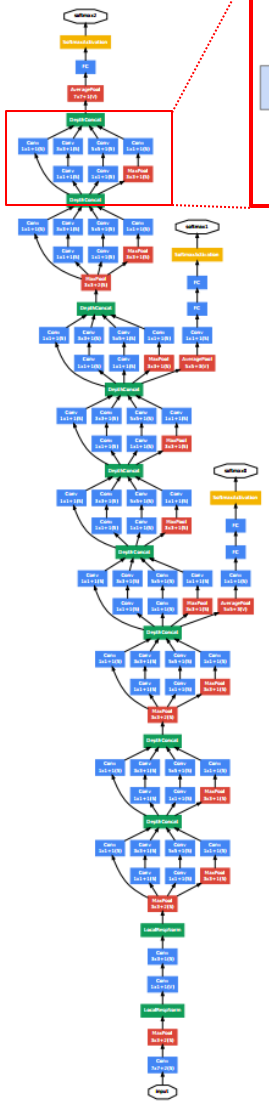
AlexNet



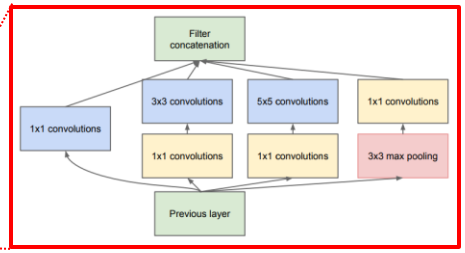
VGG



NiN



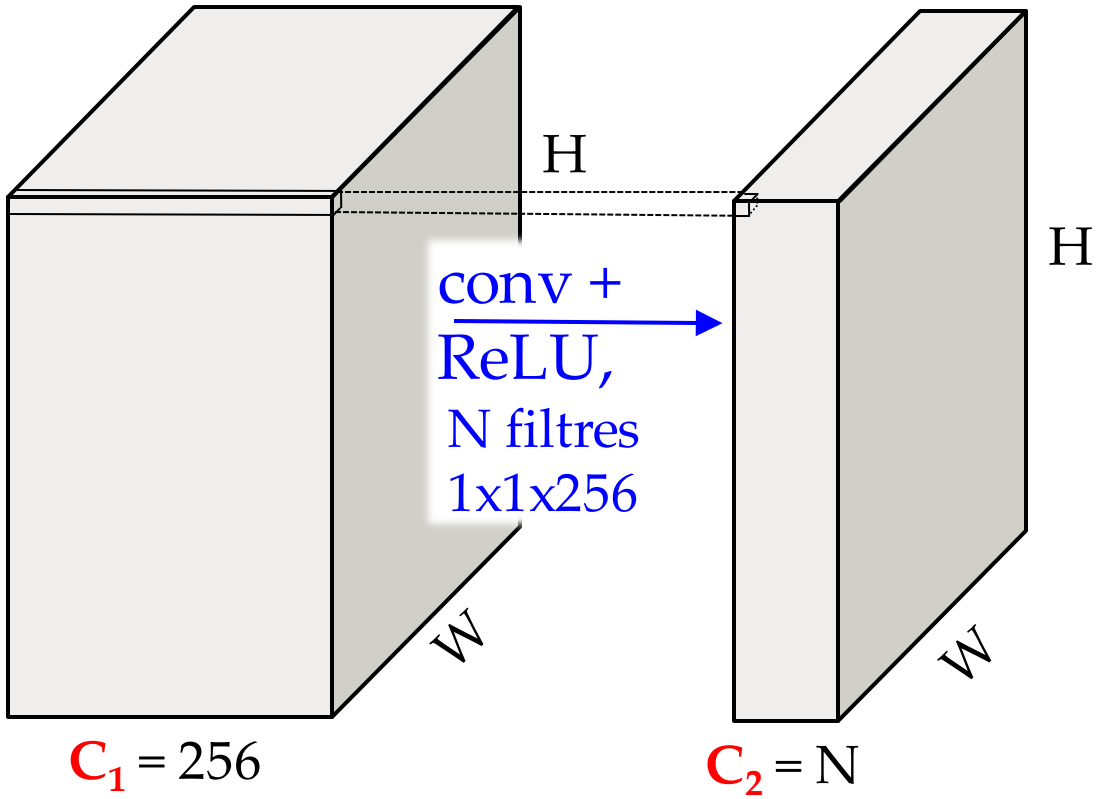
GoogLeNet



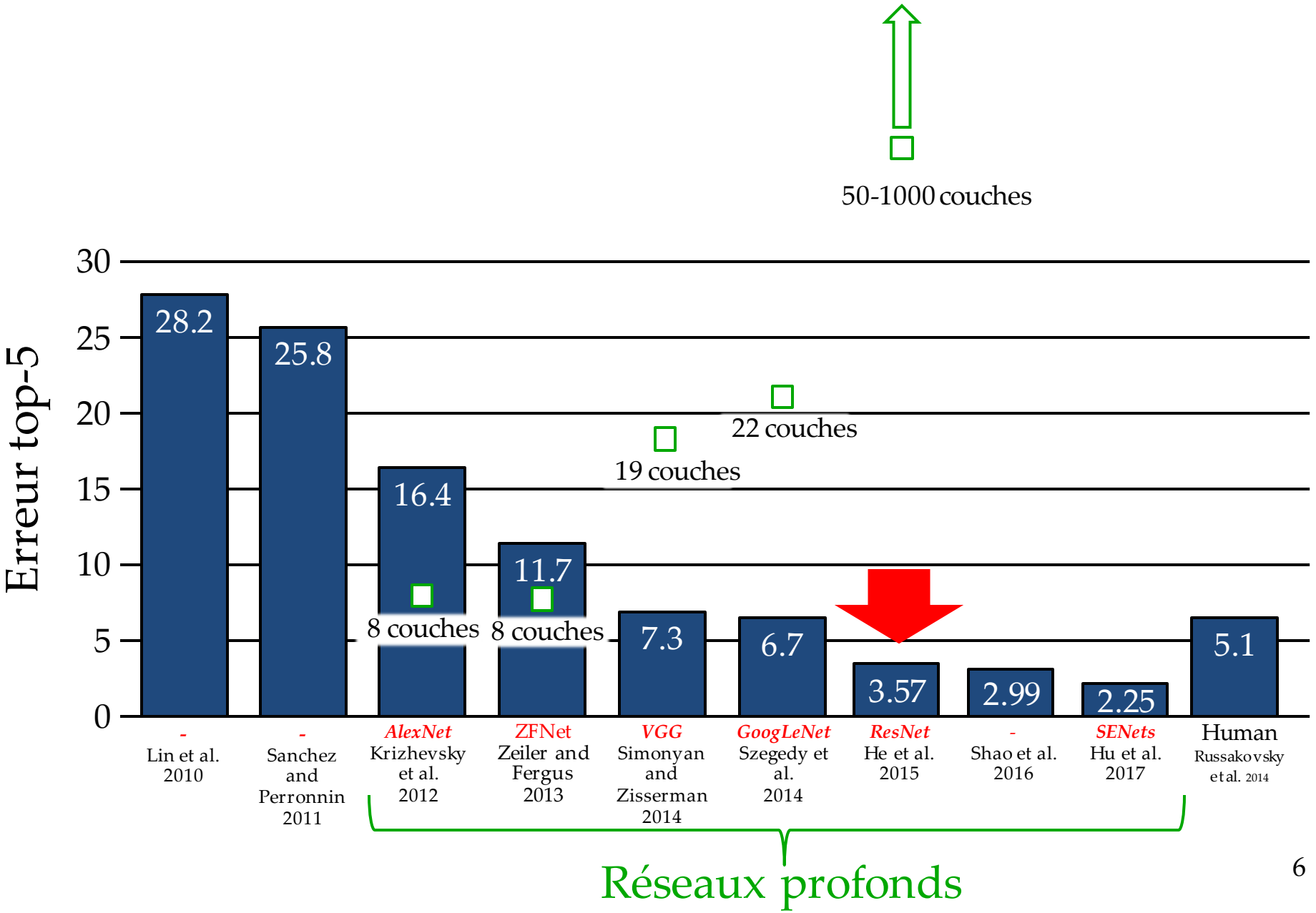
Module Inception (multi-branche)

Résumé CNN I

- conv 1x1 sont des réseaux *fully-connected*
- Servent à réduire la dimensionnalité des *feature maps*

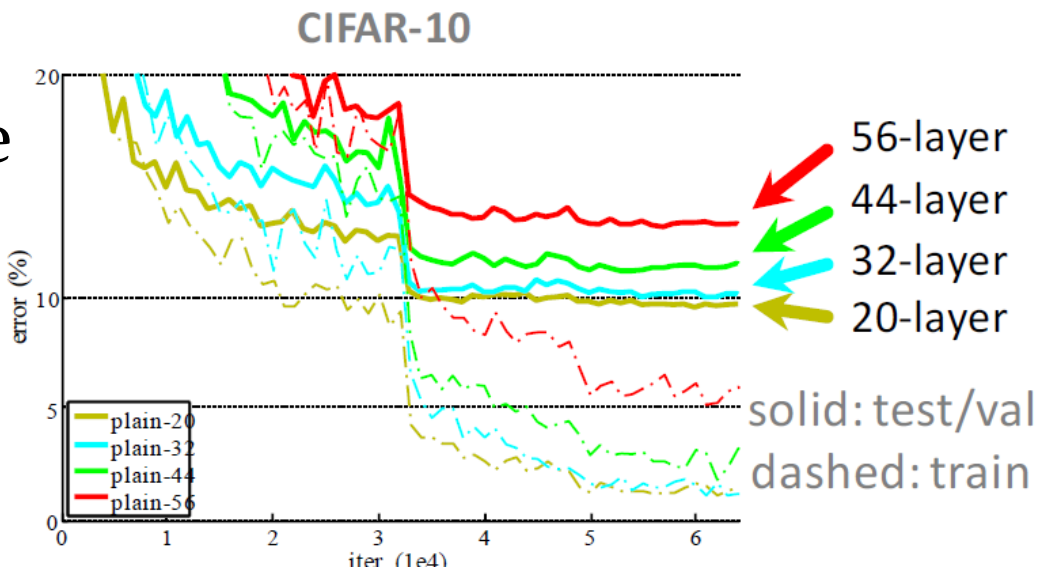


Large Scale Visual Recognition Challenge



ResNet

- Vise l'entraînement de réseaux très profonds (30+ couches)
- Problème du *vanishing gradient* en partie réglé par la Batch Norm
- Constat : dégradation des résultats passé une certaine profondeur



ICML tutorial 2016, He.

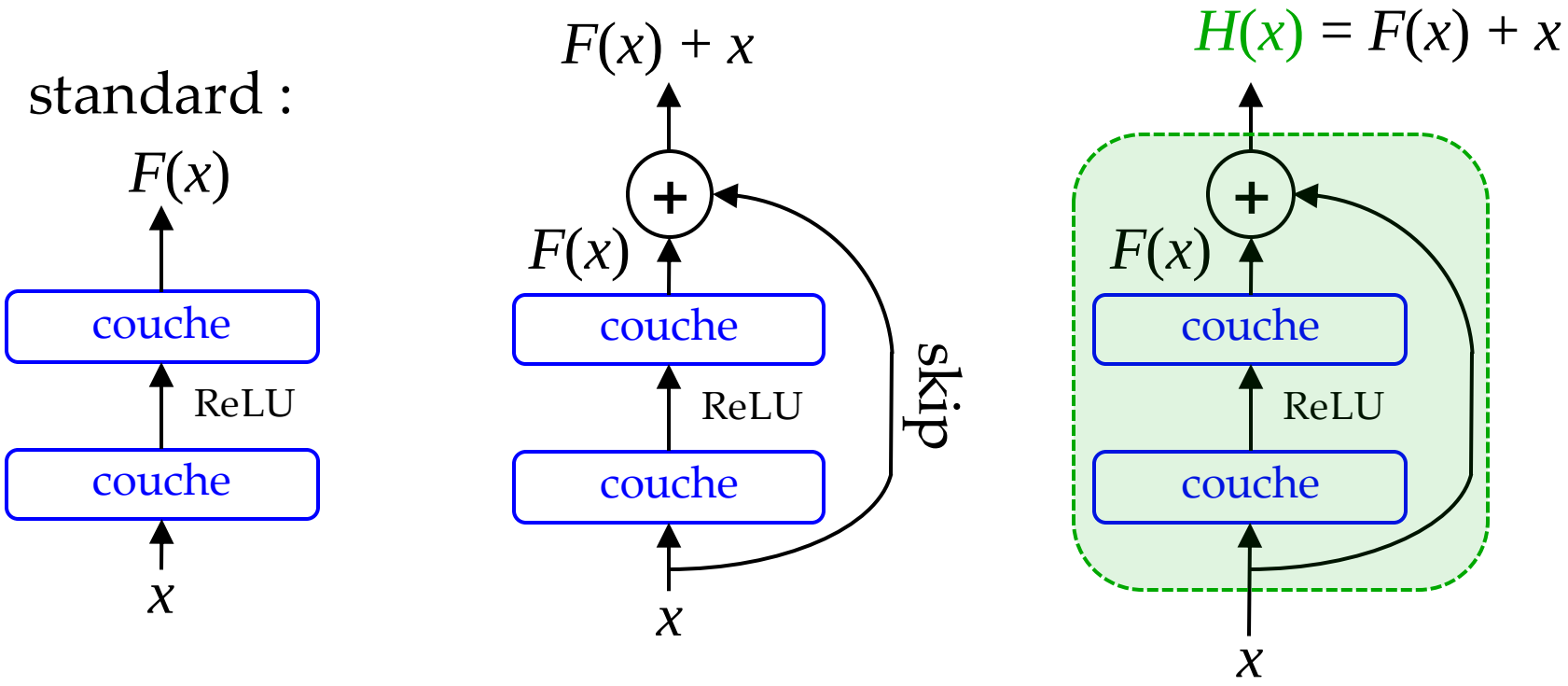
- Intuition : un réseau devrait simplement apprendre la fonction identité
  - mais l'optimisation n'y arrive pas

ResNet

- Dilution de l'information de correction (gradient)
- Difficile pour une couche de réutiliser des *features* des couches précédentes
  - perte de l'*information flow* [1]
- Difficulté d'apprendre la fonction identitaire

[1] Srivastava, R.K., Greff, K., Schmidhuber, J.: Highway networks. arXiv preprint arXiv:1505.00387 (2015)

ResNet : idée de base



$F(x)$ est le résiduel entre la fonction $H(x)$ désirée et la fonction identitaire : $F(x) = H(x) - x$

- N'ajoute aucun paramètre au réseau, très peu de calcul
- Doit avoir au moins deux couches internes

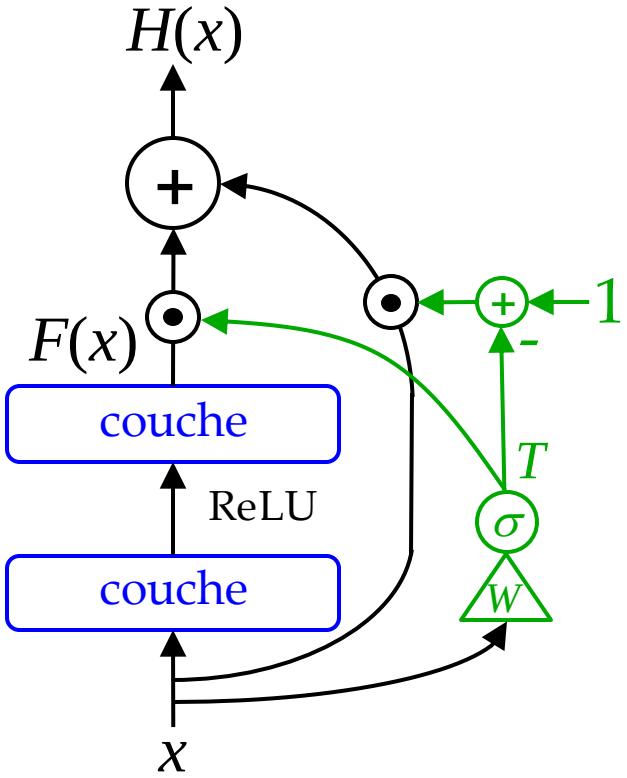
Highway network

- Compétiteur contemporain des ResNet
- Va utiliser du *gating* pour choisir le mélange résiduel vs. identité

$$H(x) = F(x)T + x(1-T)$$

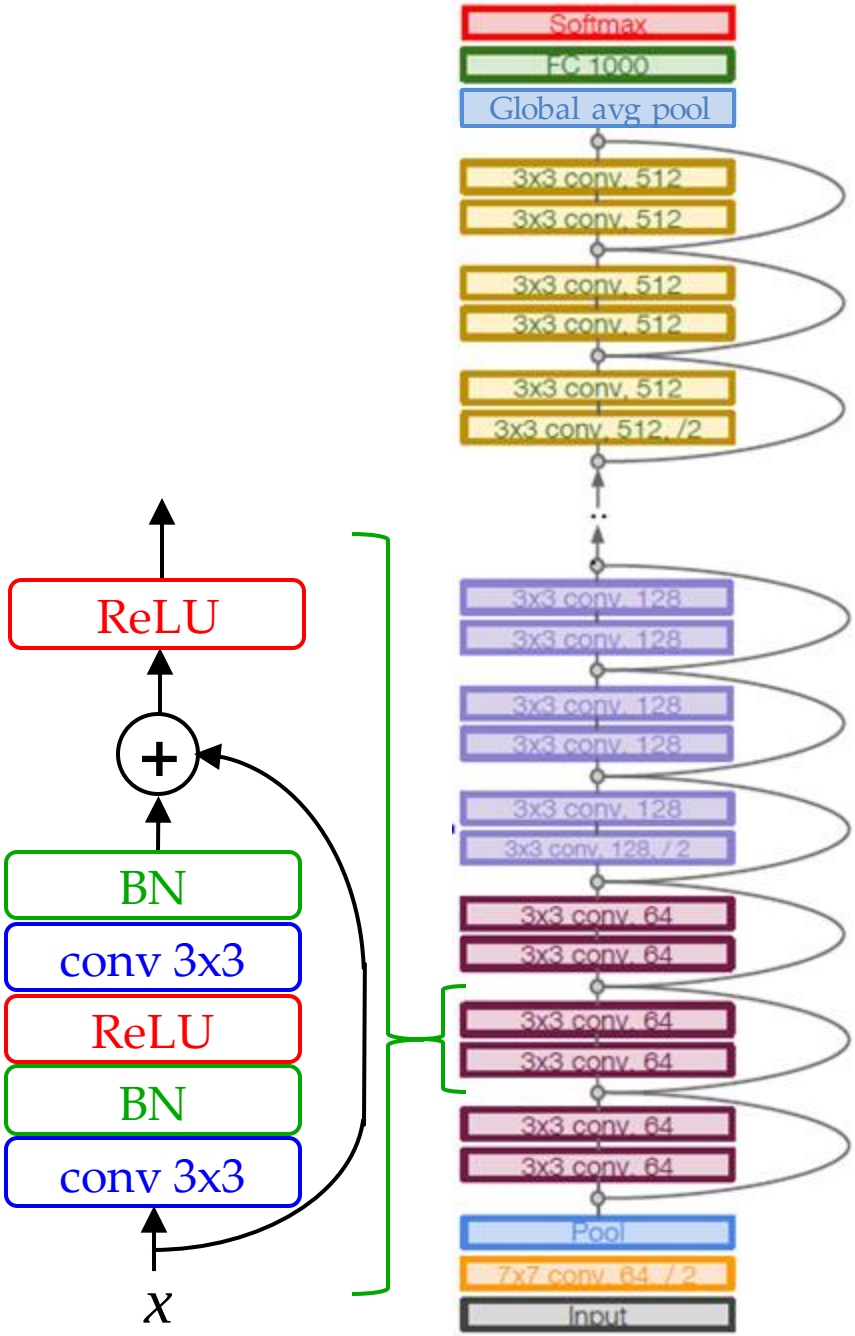
$$T(\mathbf{x}) = \sigma(\mathbf{W_T}^T\mathbf{x} + \mathbf{b_T})$$

- ResNet a fait le pari qu'il est toujours mieux de faire la somme des deux : architecture plus simple



Srivastava, R. et al., Training Very Deep Networks, NIPS 2015

ResNet



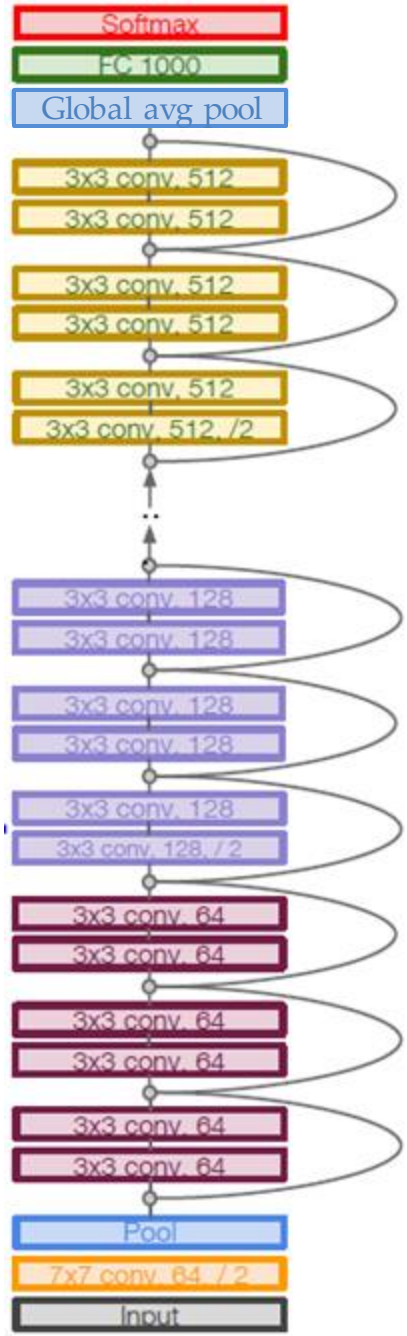
adapté de : cs231n, Université Stanford

ResNet

- Apprentissage du résiduel :
  - plus facile (car cela peut être que des petits ajustements)
  - $F(x)$ initialisé avec des petites valeurs
  - pourra facilement apprendre des mapping identitaires

Notes :

- architecture simple de conv 3x3, style VGG
- convolution 7x7 avec stride 2 à la base
- double le # filtre après chaque réduction de taille du feature map
- position des ReLU : sera différente pour version « identity mapping »
- pas de dropout



adapté de : cs231n, Université Stanford

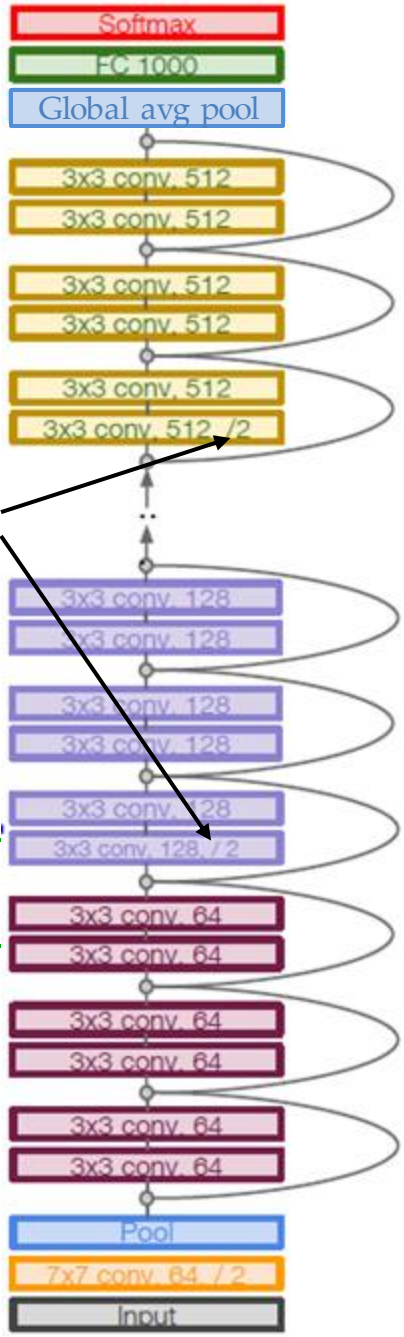
ResNet

Global Average Pooling

Downsampling se fait par des conv avec pas=2

Si besoin d'ajuster les dimensions : $y = F(x) + W_s x$

(1x1 conv)



Dans certains cas, besoin de warm up

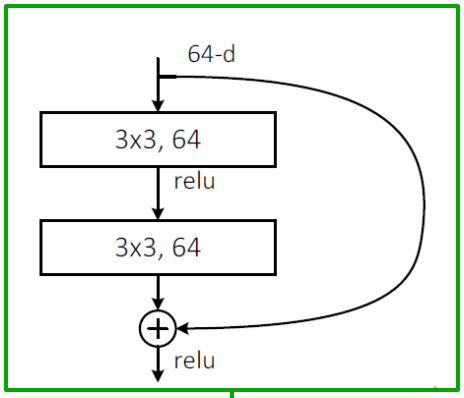
"…we find that the initial learning rate of 0.1 is slightly too large to start converging. So we use 0.01 to warm up the training until the training error is below 80% (about 400 iterations), and then go back to 0.1 and continue training."

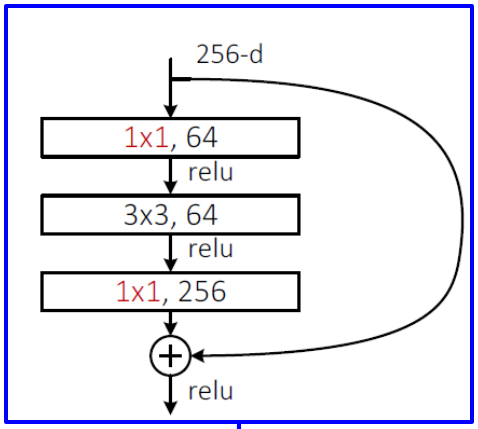
adapté de : cs231n, Université Stanford

ResNet

Bottleneck

(pour efficacité, pas pour régularisation)

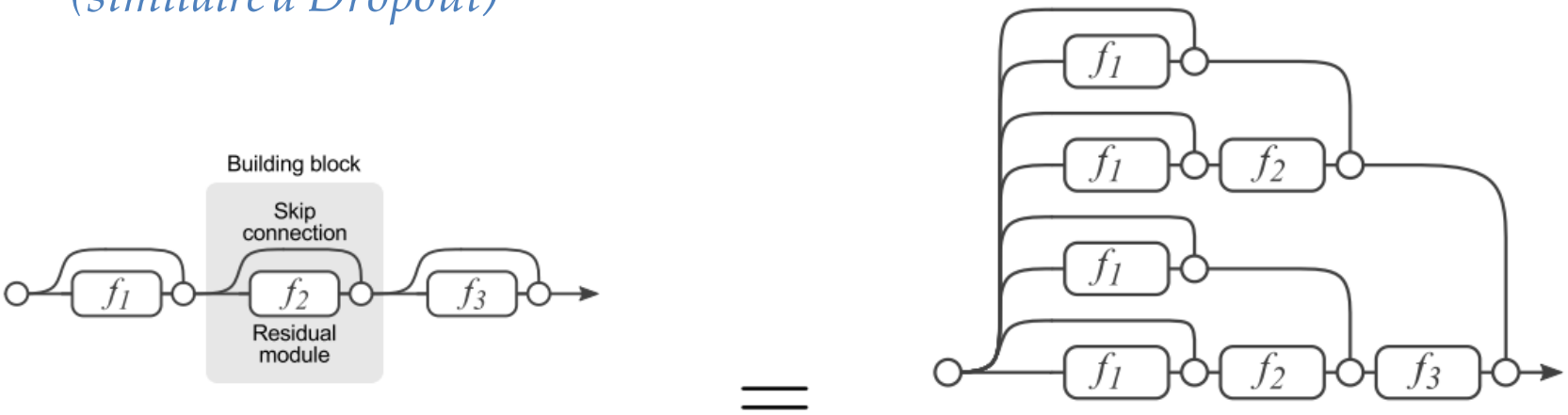




layer name	output size	18-layer	34-layer	50-layer	101-layer	152-layer
conv1	112×112	7×7, 64, stride 2				
conv2_x	56×56	3×3 max pool, stride 2				
		$\begin{bmatrix} 3\times3, 64 \\ 3\times3, 64 \end{bmatrix}\times2$	$\begin{bmatrix} 3\times3, 64 \\ 3\times3, 64 \end{bmatrix}\times3$	$\begin{bmatrix} 1\times1, 64 \\ 3\times3, 64 \\ 1\times1, 256 \end{bmatrix}\times3$	$\begin{bmatrix} 1\times1, 64 \\ 3\times3, 64 \\ 1\times1, 256 \end{bmatrix}\times3$	$\begin{bmatrix} 1\times1, 64 \\ 3\times3, 64 \\ 1\times1, 256 \end{bmatrix}\times3$
conv3_x	28×28	$\begin{bmatrix} 3\times3, 128 \\ 3\times3, 128 \end{bmatrix}\times2$	$\begin{bmatrix} 3\times3, 128 \\ 3\times3, 128 \end{bmatrix}\times4$	$\begin{bmatrix} 1\times1, 128 \\ 3\times3, 128 \\ 1\times1, 512 \end{bmatrix}\times4$	$\begin{bmatrix} 1\times1, 128 \\ 3\times3, 128 \\ 1\times1, 512 \end{bmatrix}\times4$	$\begin{bmatrix} 1\times1, 128 \\ 3\times3, 128 \\ 1\times1, 512 \end{bmatrix}\times8$
conv4_x	14×14	$\begin{bmatrix} 3\times3, 256 \\ 3\times3, 256 \end{bmatrix}\times2$	$\begin{bmatrix} 3\times3, 256 \\ 3\times3, 256 \end{bmatrix}\times6$	$\begin{bmatrix} 1\times1, 256 \\ 3\times3, 256 \\ 1\times1, 1024 \end{bmatrix}\times6$	$\begin{bmatrix} 1\times1, 256 \\ 3\times3, 256 \\ 1\times1, 1024 \end{bmatrix}\times23$	$\begin{bmatrix} 1\times1, 256 \\ 3\times3, 256 \\ 1\times1, 1024 \end{bmatrix}\times36$
conv5_x	7×7	$\begin{bmatrix} 3\times3, 512 \\ 3\times3, 512 \end{bmatrix}\times2$	$\begin{bmatrix} 3\times3, 512 \\ 3\times3, 512 \end{bmatrix}\times3$	$\begin{bmatrix} 1\times1, 512 \\ 3\times3, 512 \\ 1\times1, 2048 \end{bmatrix}\times3$	$\begin{bmatrix} 1\times1, 512 \\ 3\times3, 512 \\ 1\times1, 2048 \end{bmatrix}\times3$	$\begin{bmatrix} 1\times1, 512 \\ 3\times3, 512 \\ 1\times1, 2048 \end{bmatrix}\times3$
	1×1	average pool, 1000-d fc, softmax				
FLOPs		1.8×10^9	3.6×10^9	3.8×10^9	7.6×10^9	11.3×10^9

ResNet : ensemble implicite

- Ensemble exponentiel 2^L de réseaux
  (similaire à Dropout)



(a) Conventional 3-block residual network

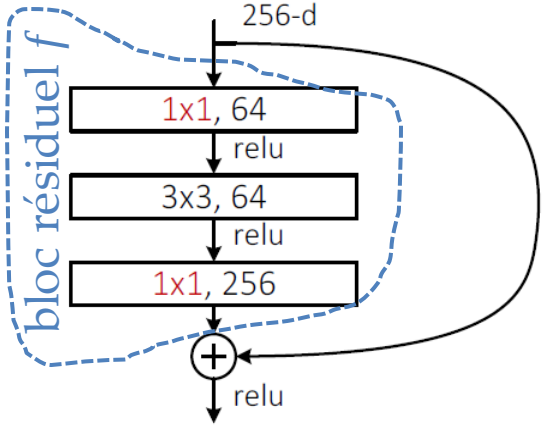
(b) Unraveled view of (a)

- Gradient est atténué dans le résiduel :
  - profondeur effective du gradient est de 10-34 couches (sur 110)

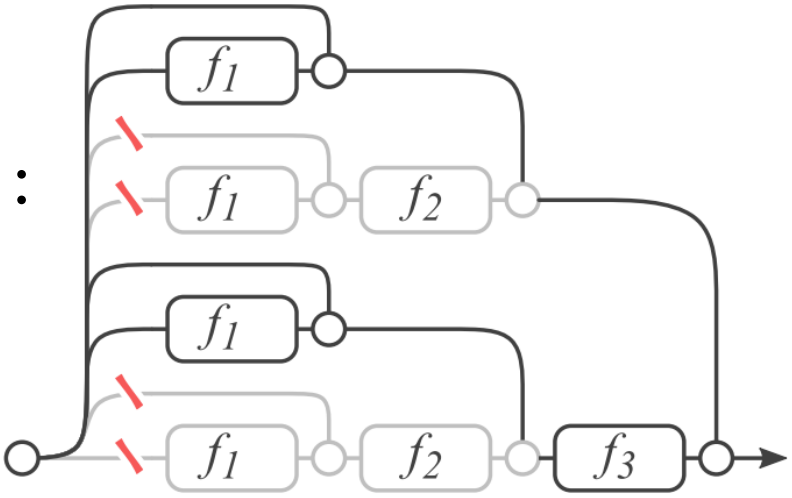
A. Veit, M. Wilber and S. Belongie. Residual Networks Behave Like Ensembles of Relatively Shallow Networks. arXiv:1605.06431v2,2016.

ResNet : ensemble implicite

- Si l'on retire un bloc résiduel ResNet, on élimine un nombre de sous-réseaux
- Il reste encore 2^{L-1} sous-réseaux



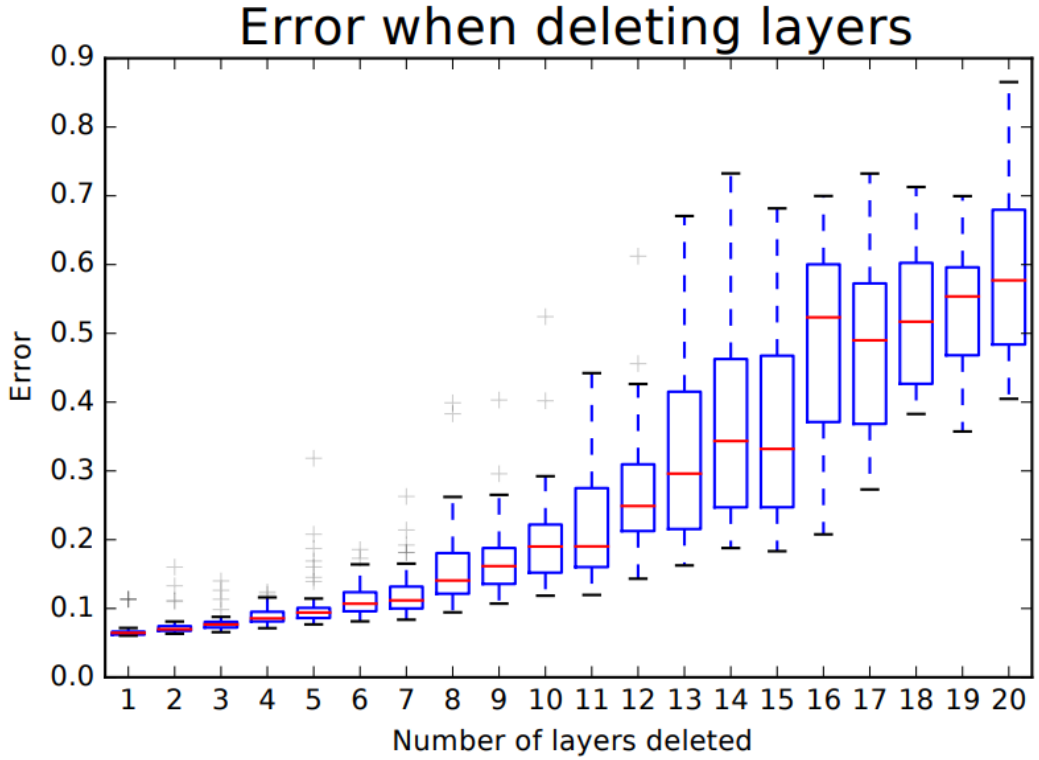
On retire f_2 :



A. Veit, M. Wilber and S. Belongie. Residual Networks Behave Like Ensembles of Relatively Shallow Networks. arXiv:1605.06431v2,2016.

ResNet : ensemble implicite

- Retirer des couches en testing pour ResNet

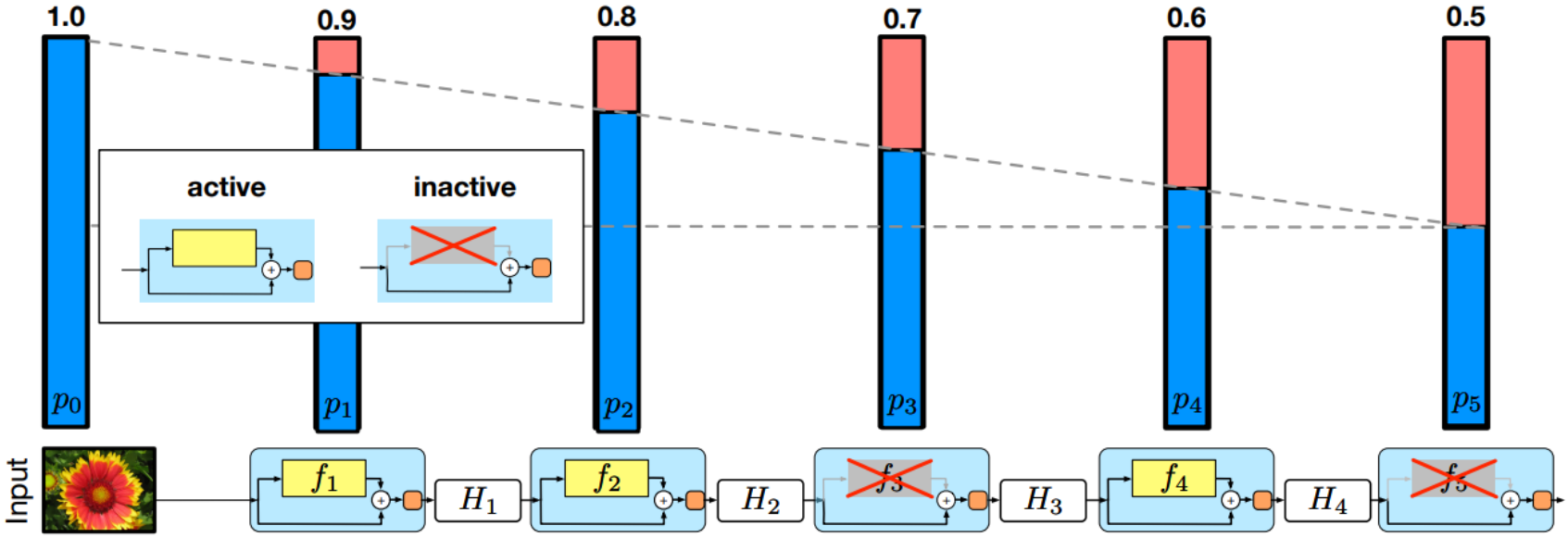


- Catastrophique pour VGG!

A. Veit, M. Wilber and S. Belongie. Residual Networks Behave Like Ensembles of Relatively Shallow Networks. arXiv:1605.06431v2,2016.

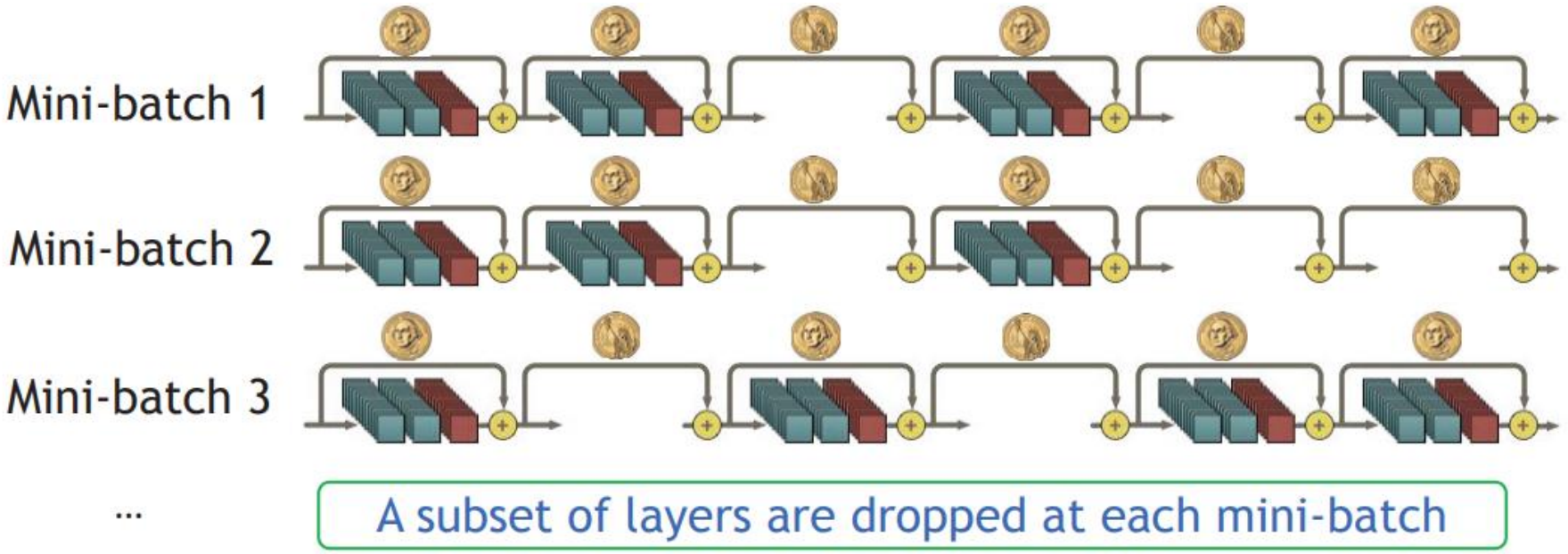
Stochastic Depth

- Aléatoirement retirer des blocs résiduels lors du training
  - dropout empiriquement inutile selon eux
- Conserver plus souvent les blocs résiduels de bas niveau



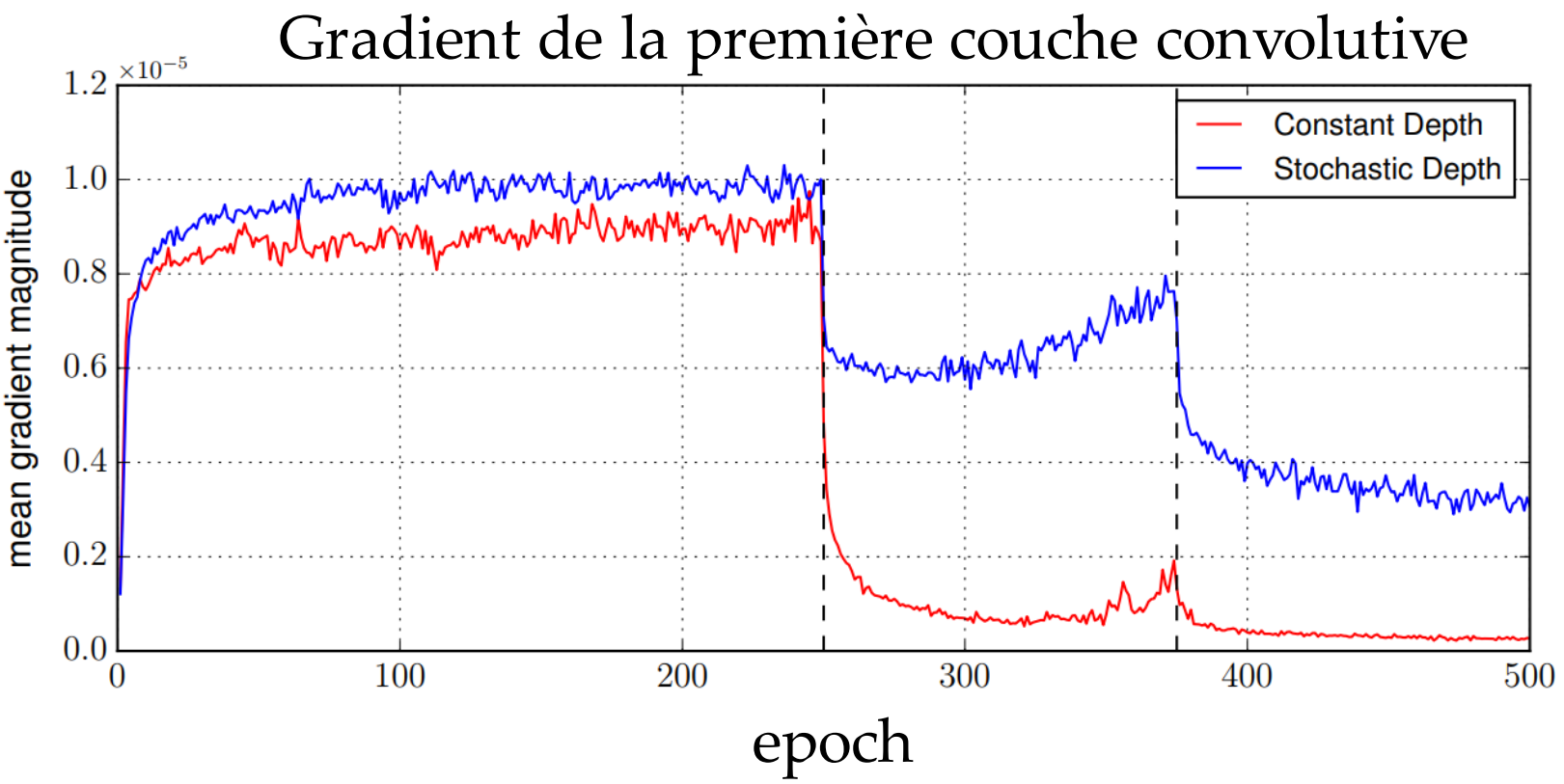
Huang et al. Deep Network with Stochastic Depth, 2016.

Stochastic Depth



Stochastic Depth

- Améliore le flot du gradient, en réduisant le nombre de couches



Huang et al. Deep Network with Stochastic Depth, 2016.

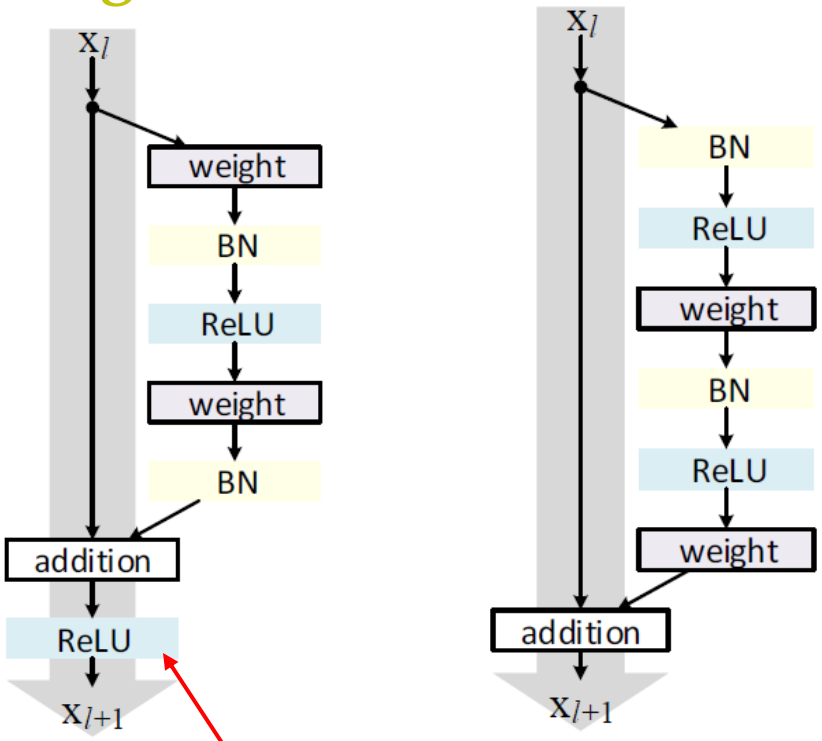
Stochastic Depth

- Accélère l'entraînement :
  - réseaux moins profonds sont plus rapides à entraîner
  - 25% moins de calcul (si décroissance linéaire 1→0.5 pour probabilité p_l droper le bloc l)
- Forme de régularisation
- Utilise le plein réseau en test
  - calibrer les forces des features en fonction de p_l
  - comme avec dropout

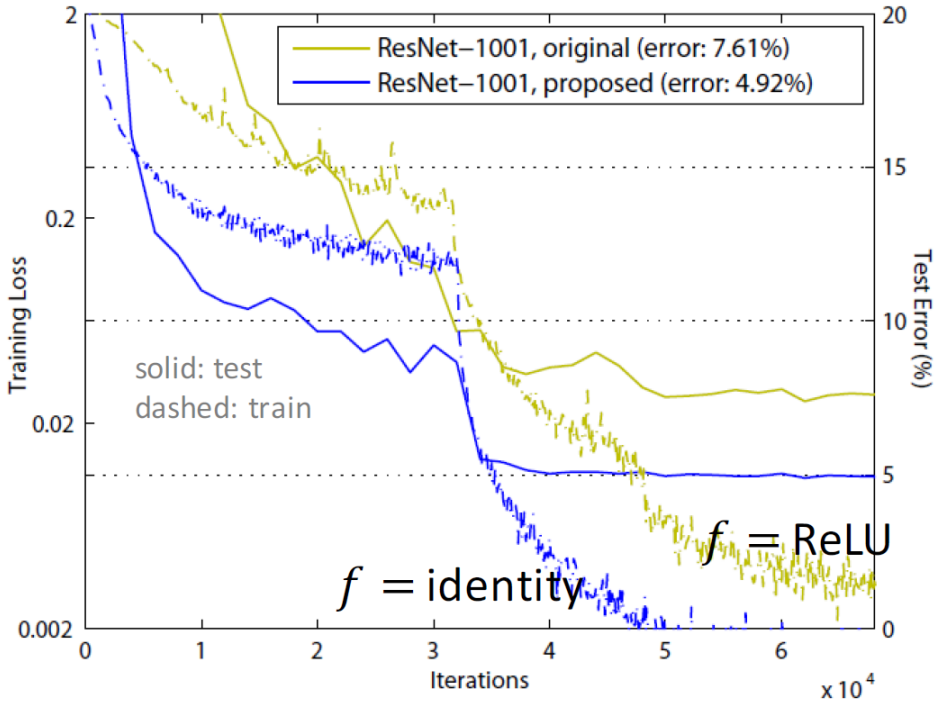
Huang et al. Deep Network with Stochastic Depth, 2016.

ResNet version preactivation

Original **«pre-activation»**



ReLU dans le chemin du gradient



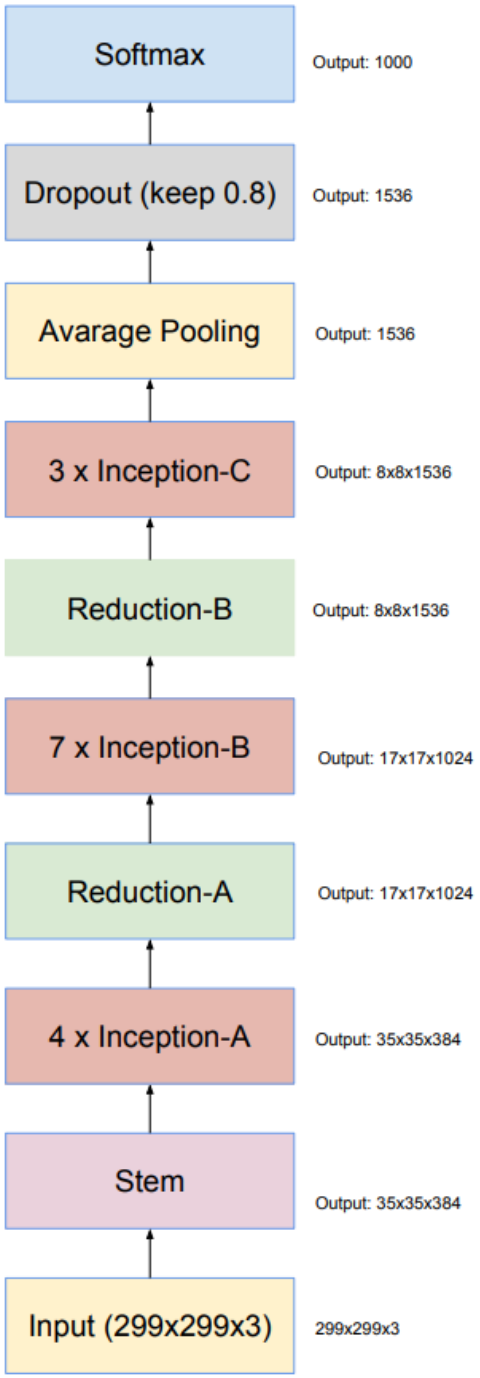
tiré : ICML workshop 2016, He.

- Bloc résiduel amélioré
- Meilleur flot du gradient
- Améliore les résultats
  - 6.7% → 4.8% top-5

He et al., Identity Mappings in Deep Residual Networks, ECCV 2016.

Variations de ResNet

Inception v4

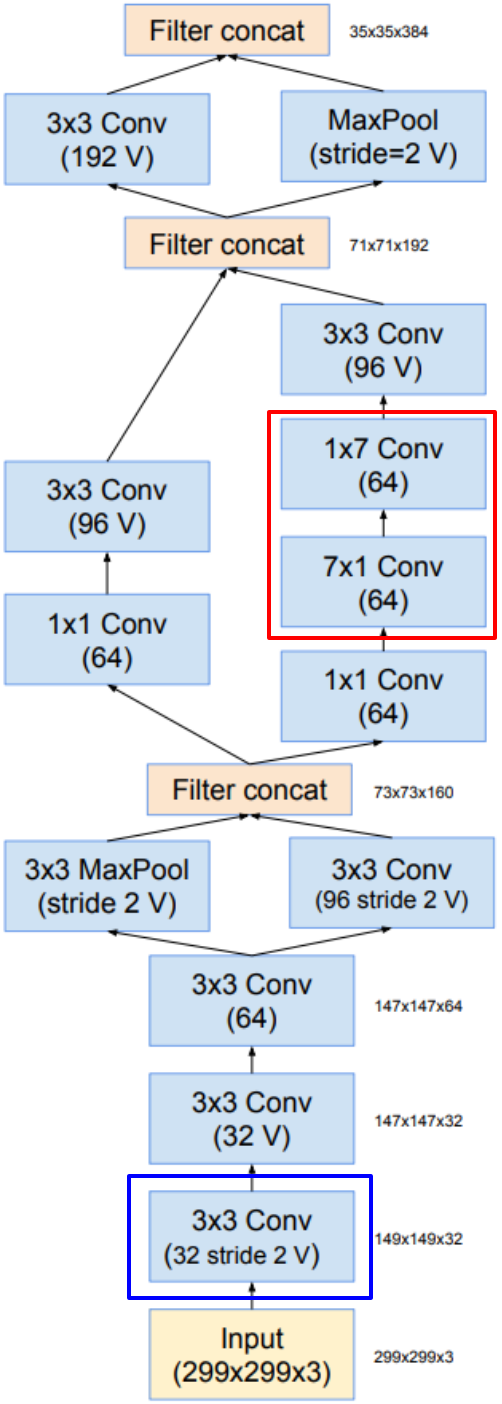


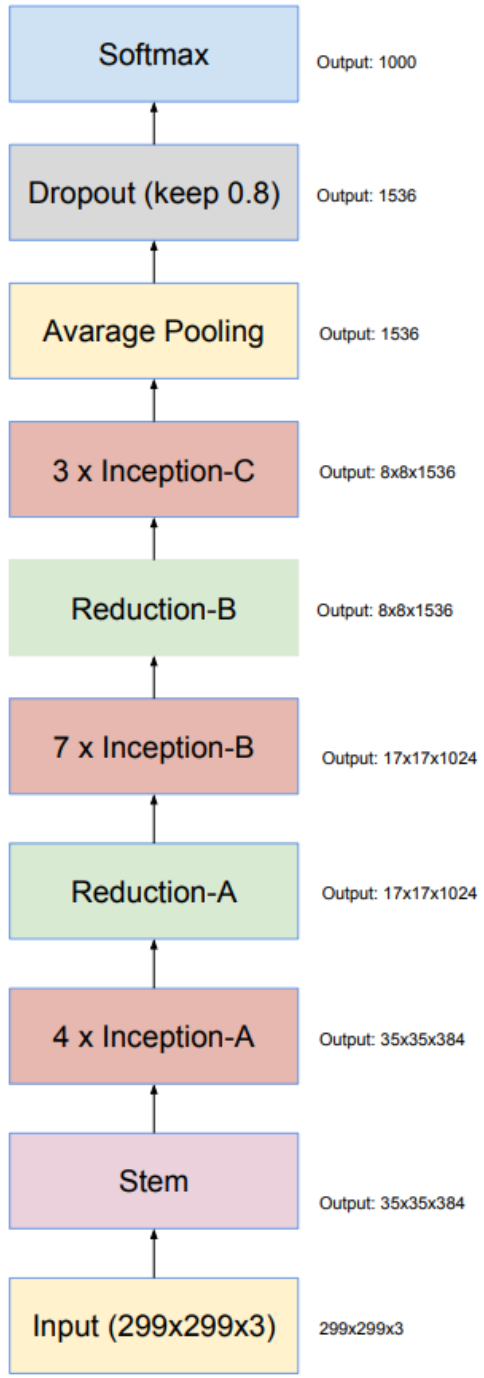
Incepti

- convolutions 1x7 et 7x1 (introduits dans v2 et v3)

Champ récepteur de 7x7, mais avec 14/49ème param.

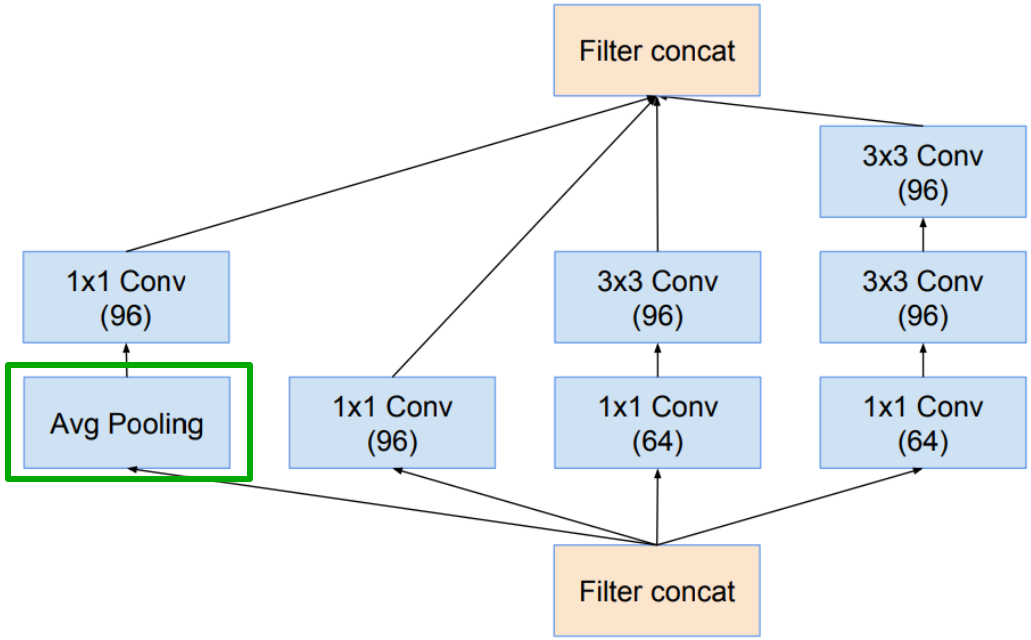
- Base a seulement un convolution 3x3, stride 2

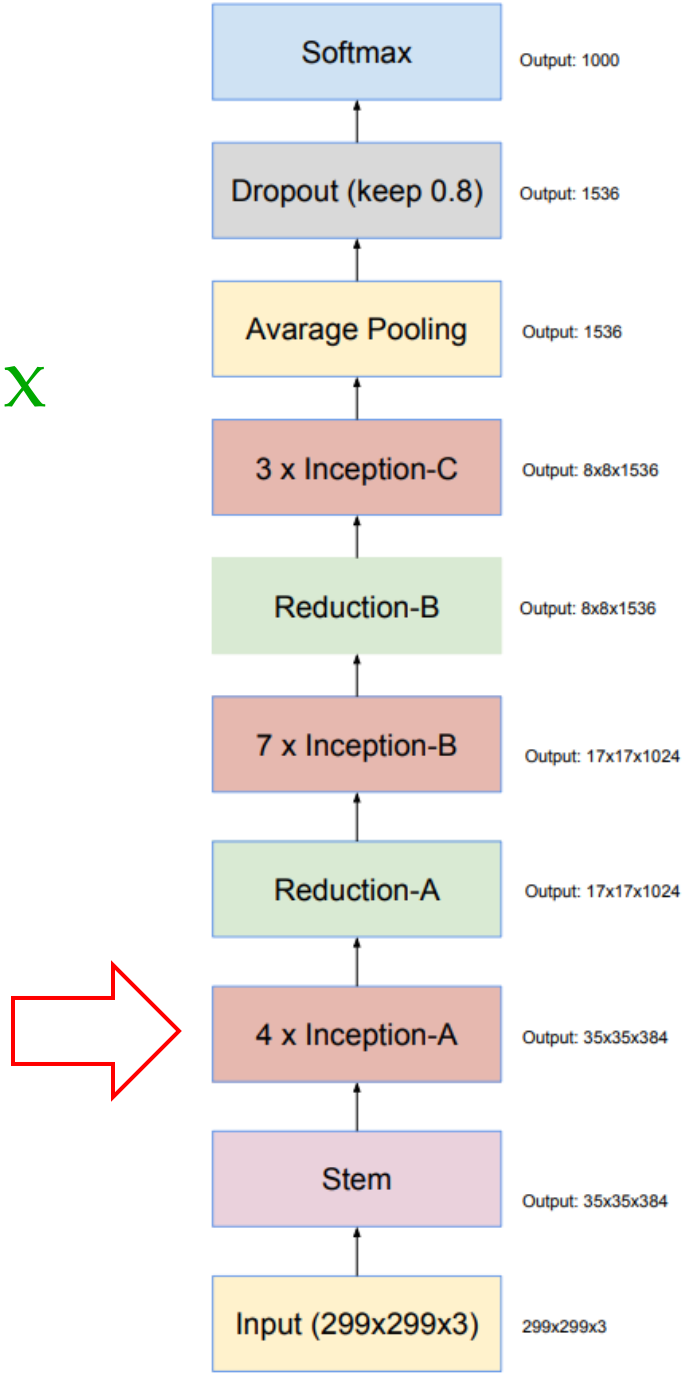




Inception v4

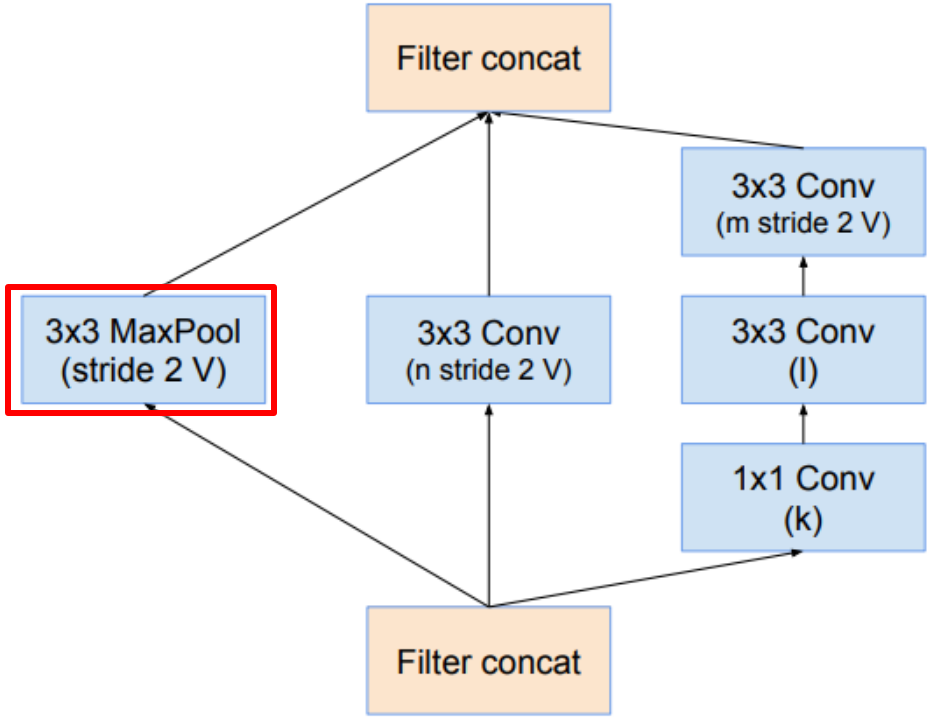
- Les 5x5 ont disparus
- Avg Pooling au lieu de max

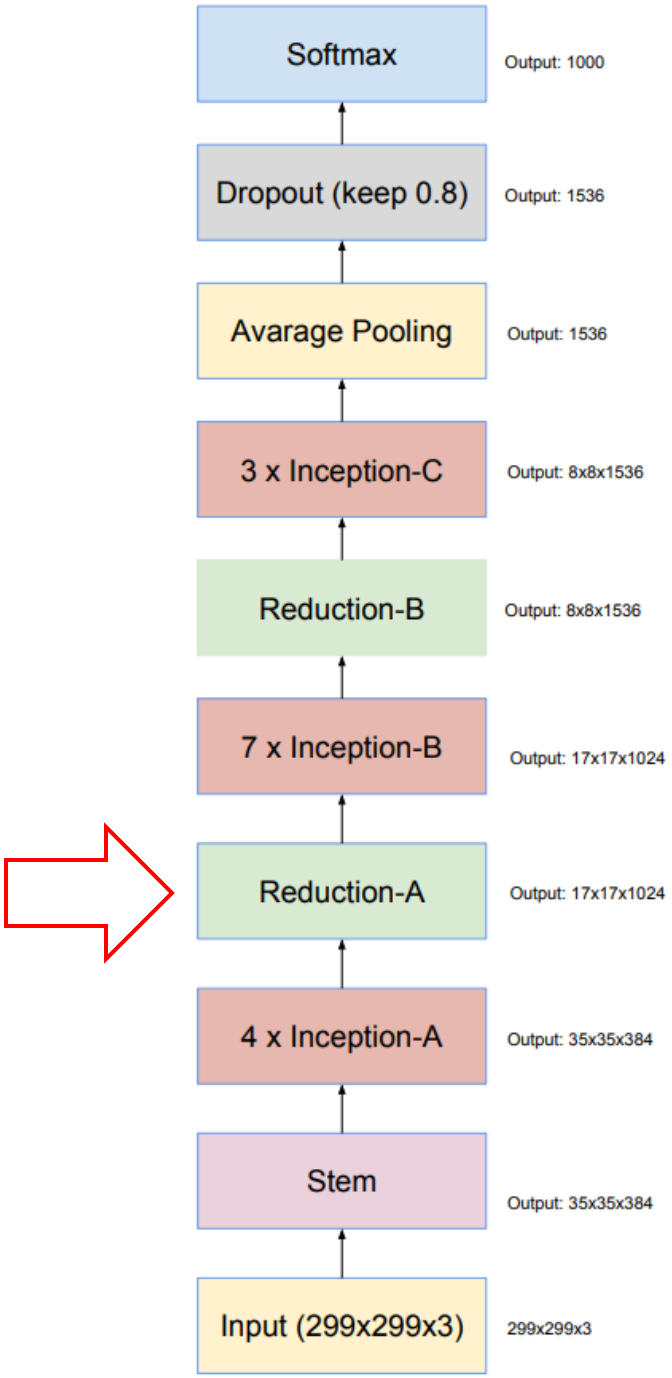




Inception v4

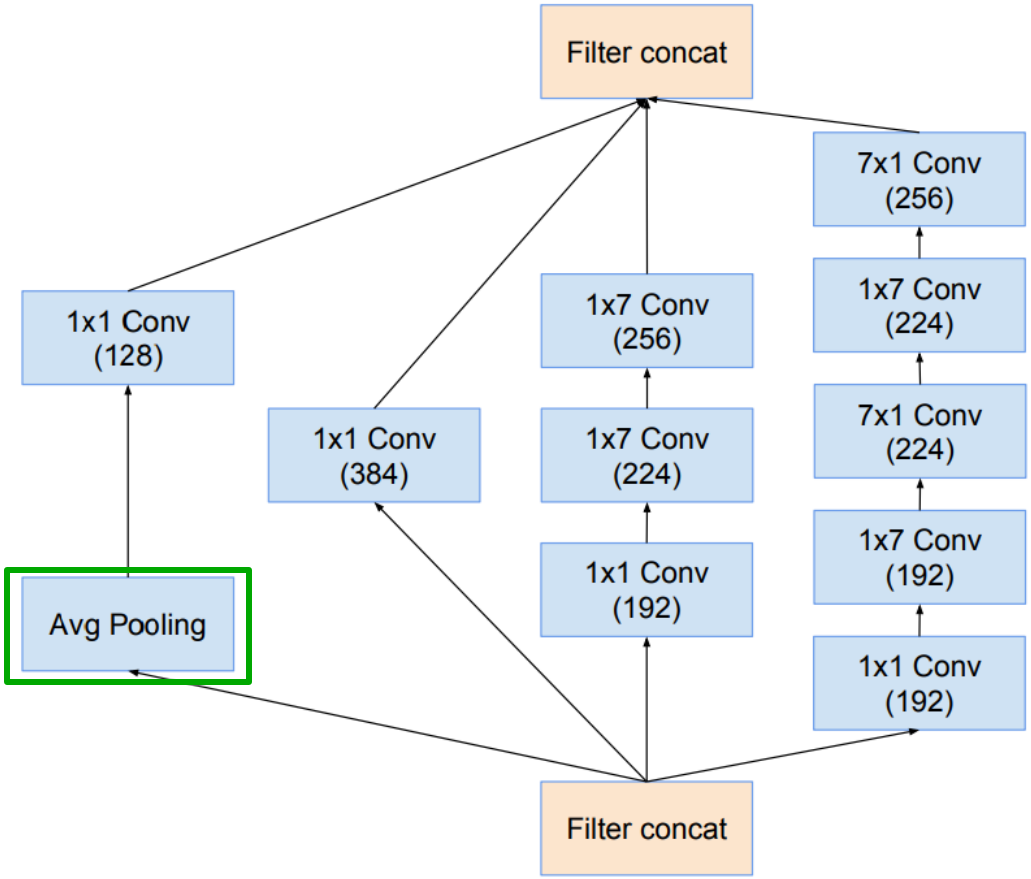
- Max Pool

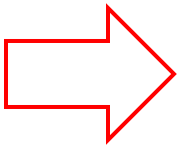


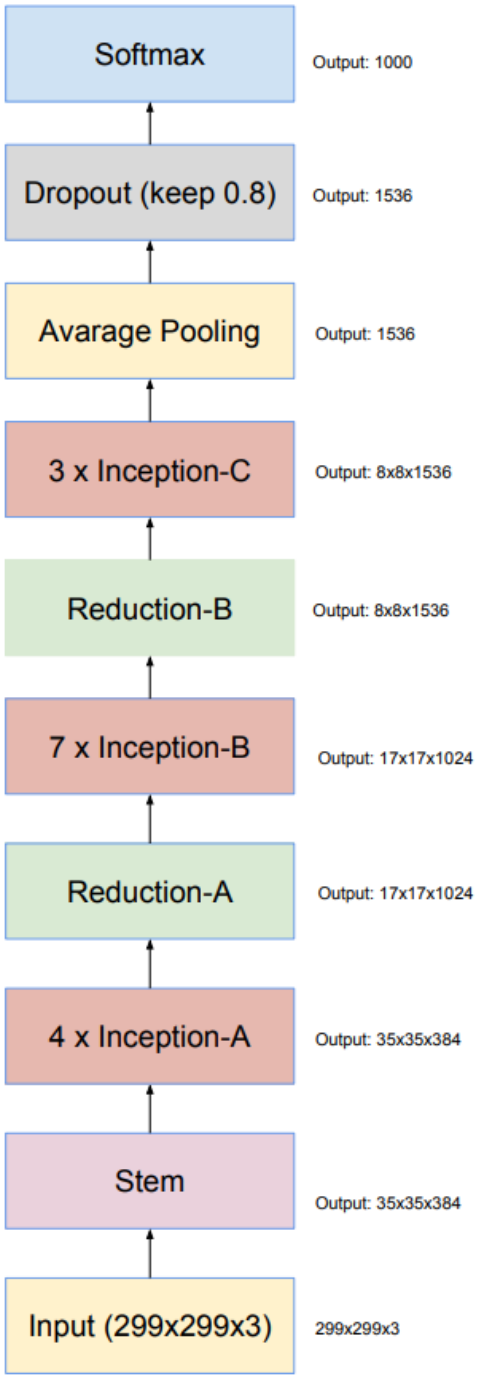


Inception v4

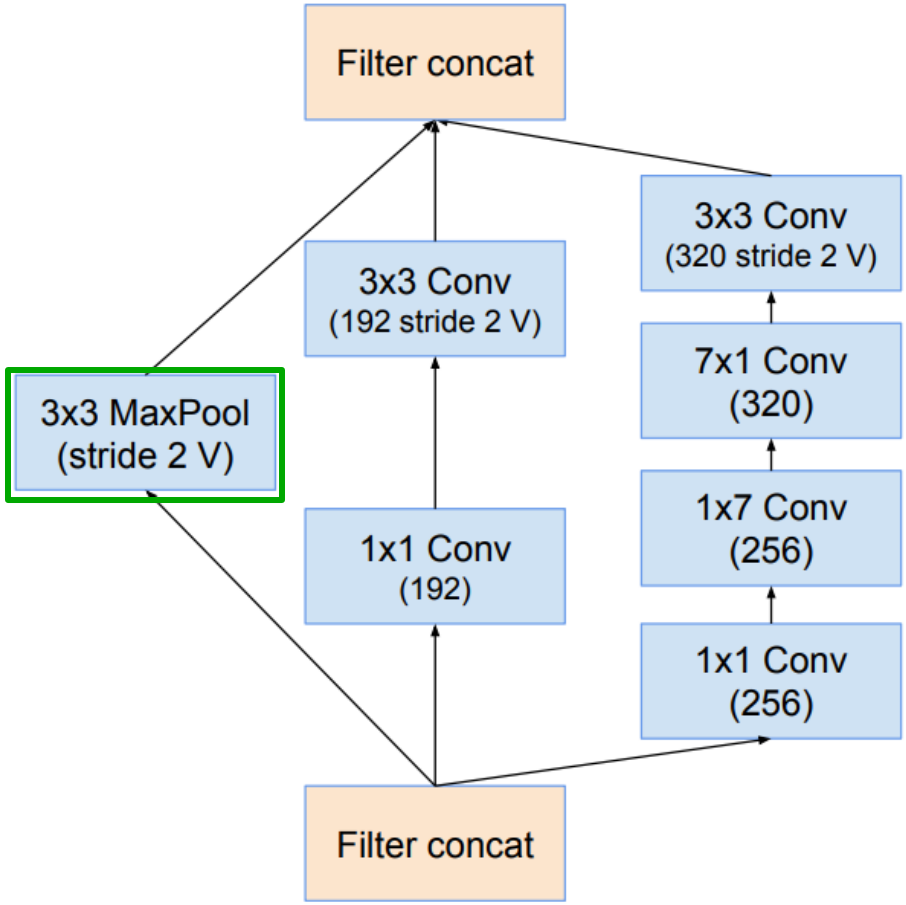
- 7x1 et 1x7 conv

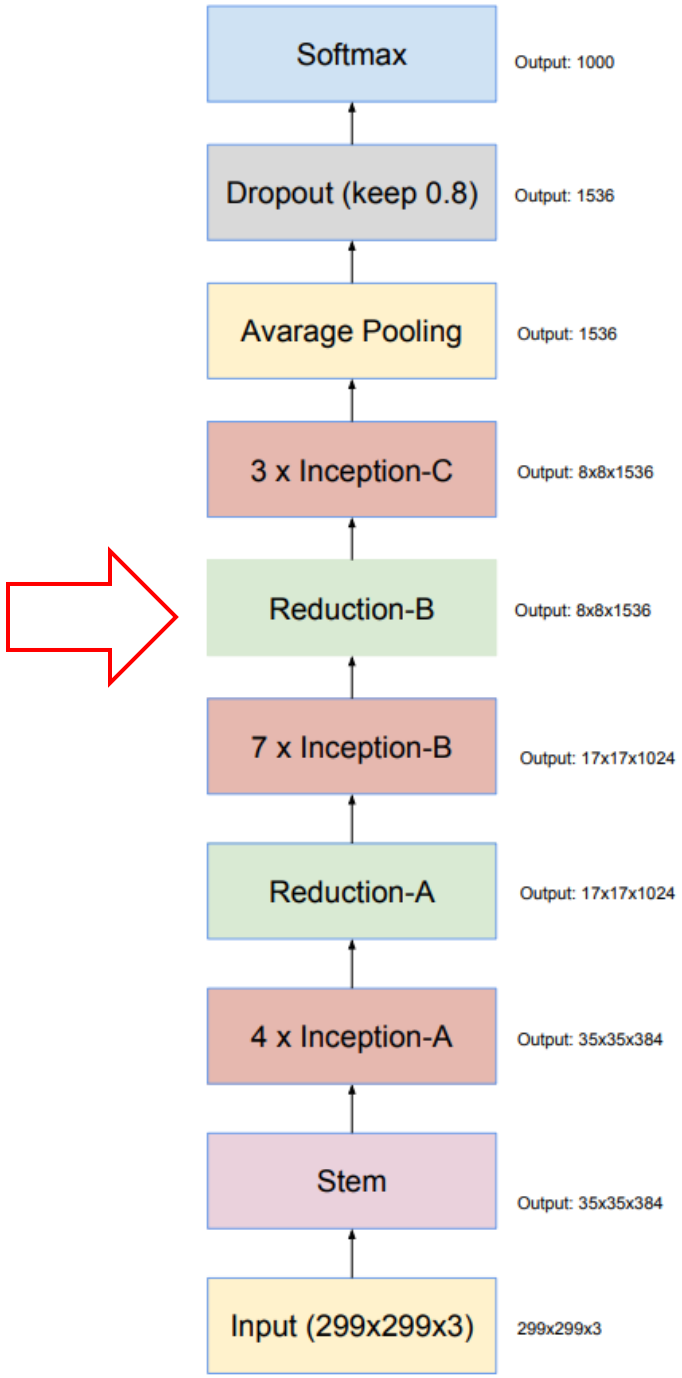






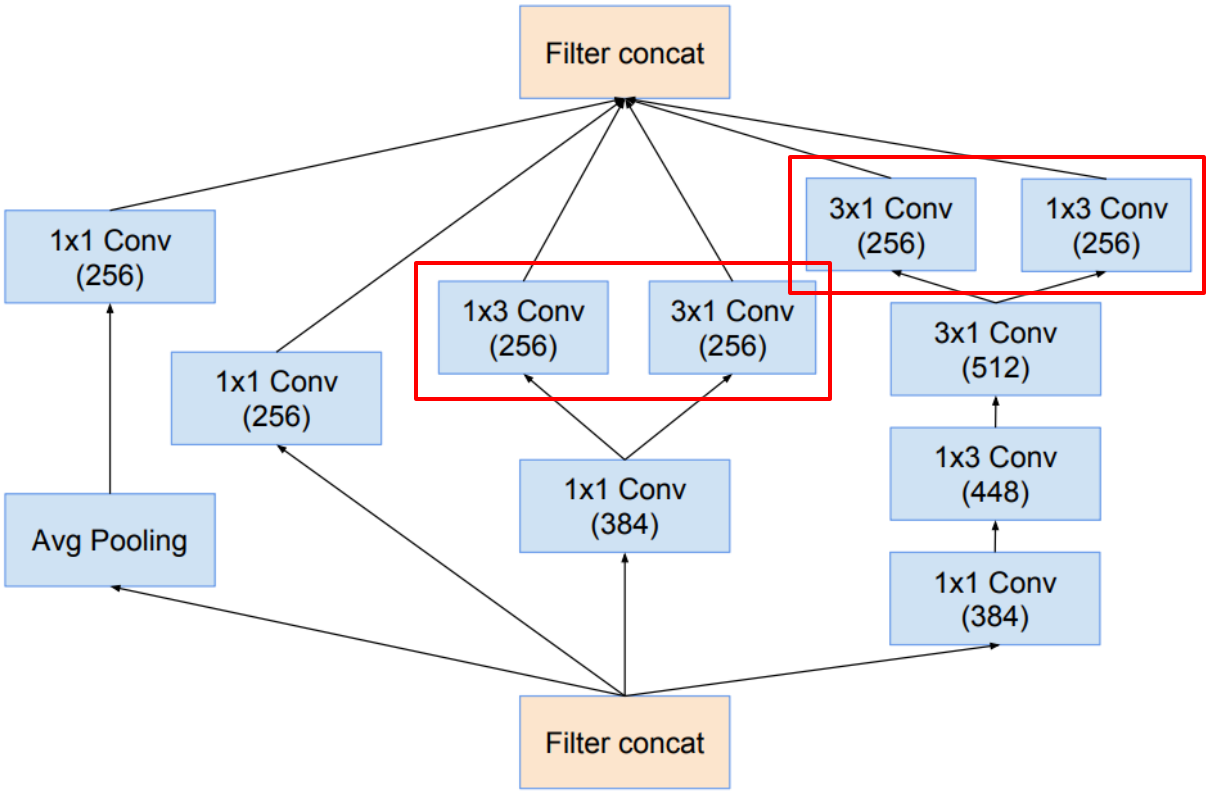
Inception v4

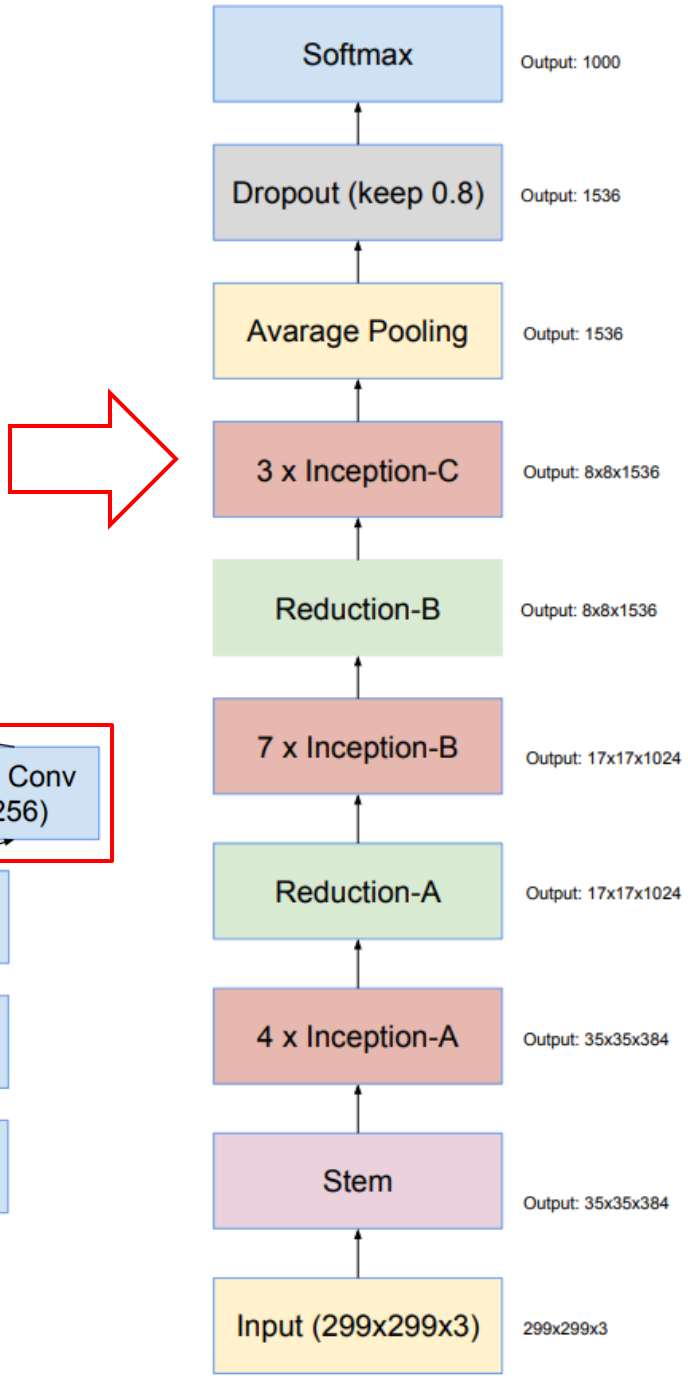




Inception v4

- 3x1 et 1x3 en parallèle

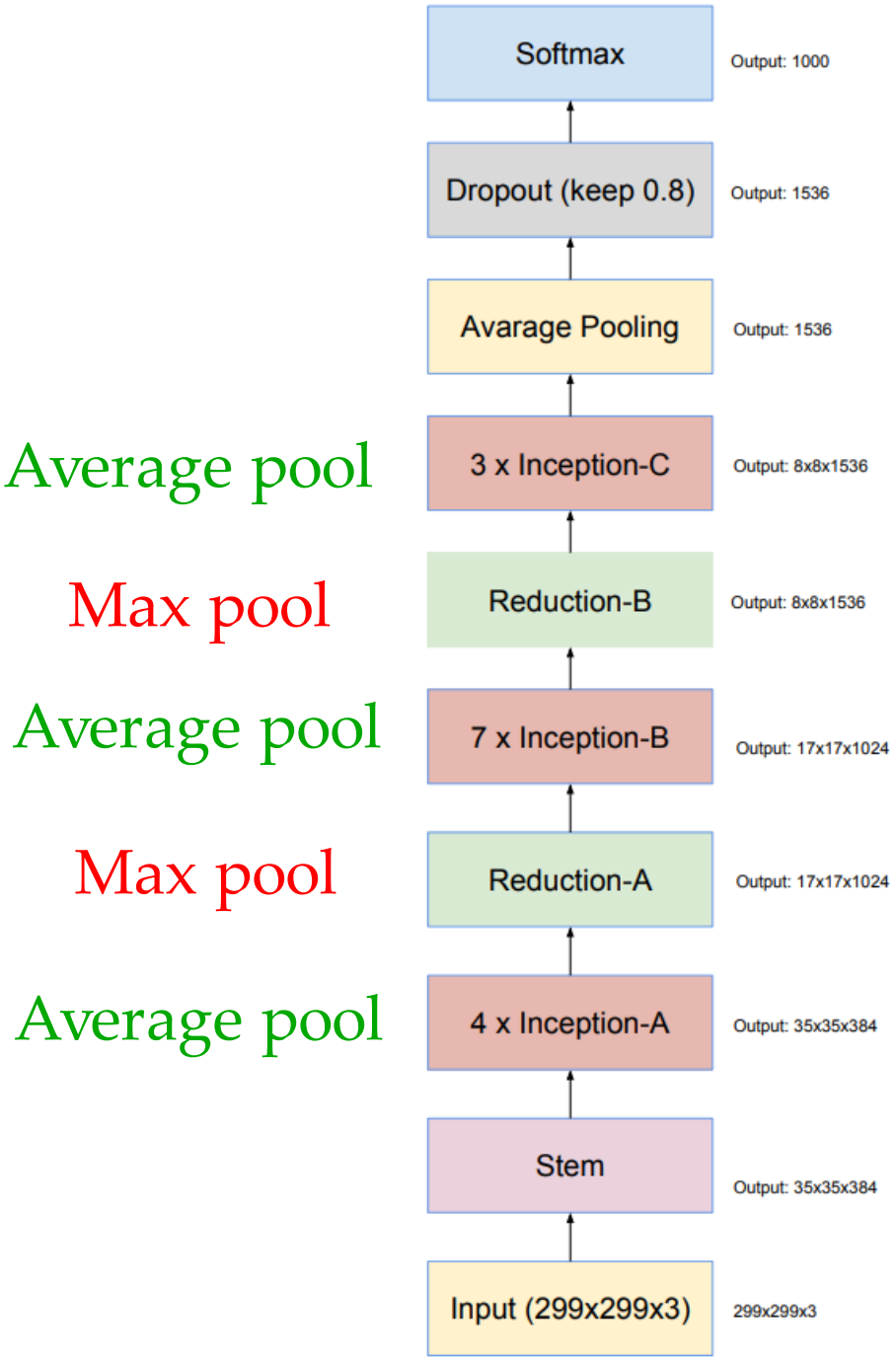




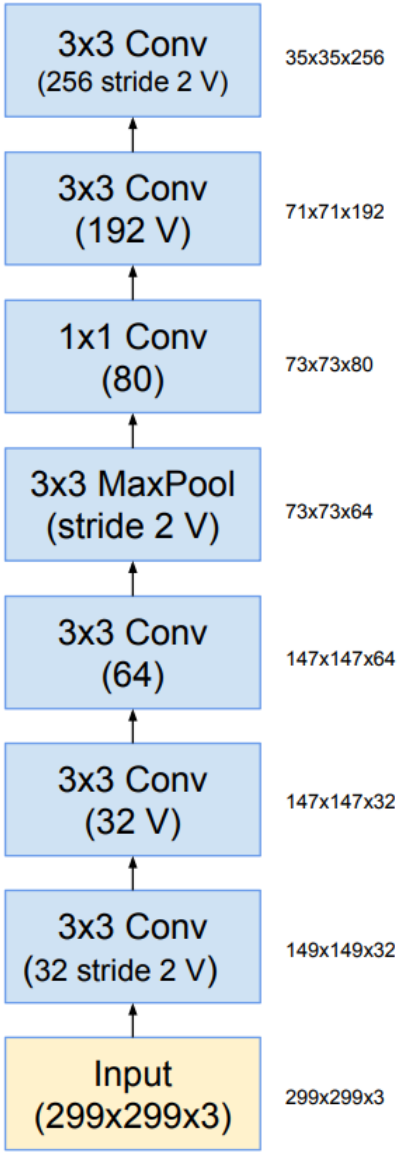
Inception v4

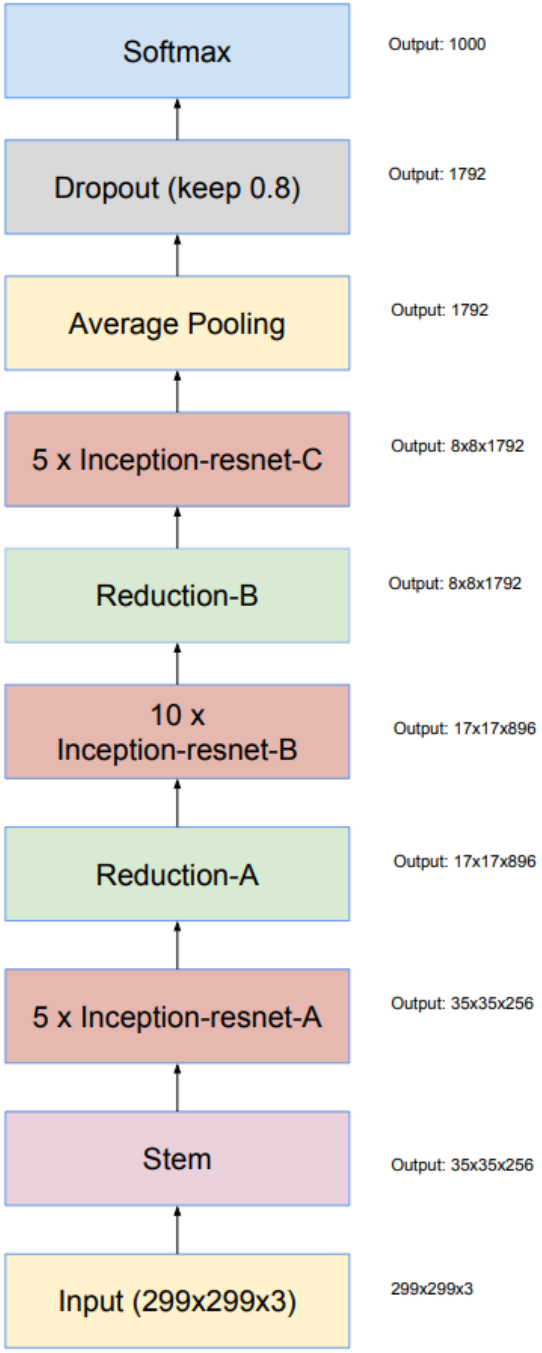
Différences avec GoogLeNet

- Couches de réduction plus complexes
- Convolutions avec *stride*≠1
- Aucun 5x5
- 1x7 et 7x1
- 1x3 et 3x1 :
  - en série
  - en parallèle
- etc…



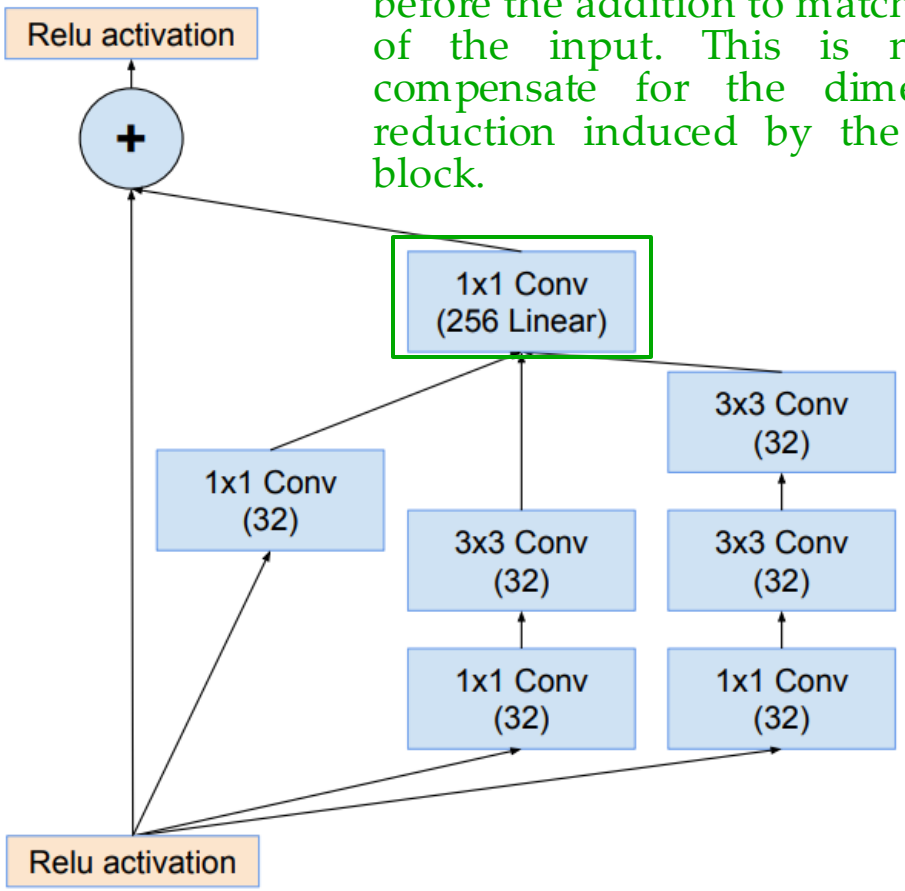
Inception-ResNet v1

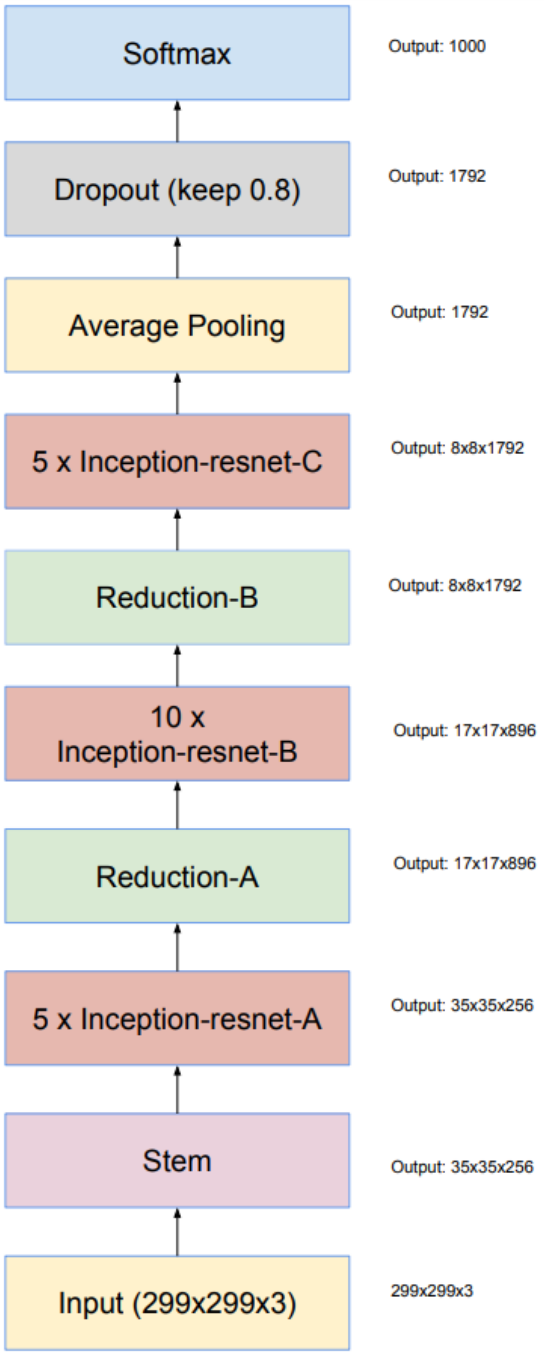




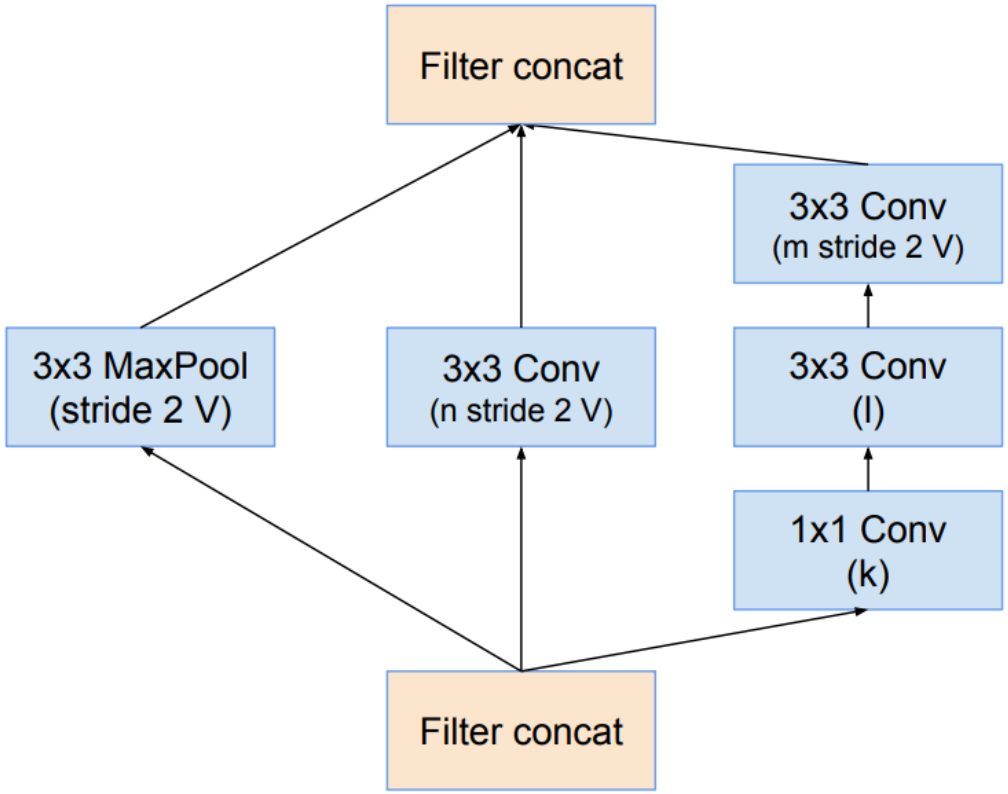
Inception-ResNet v1

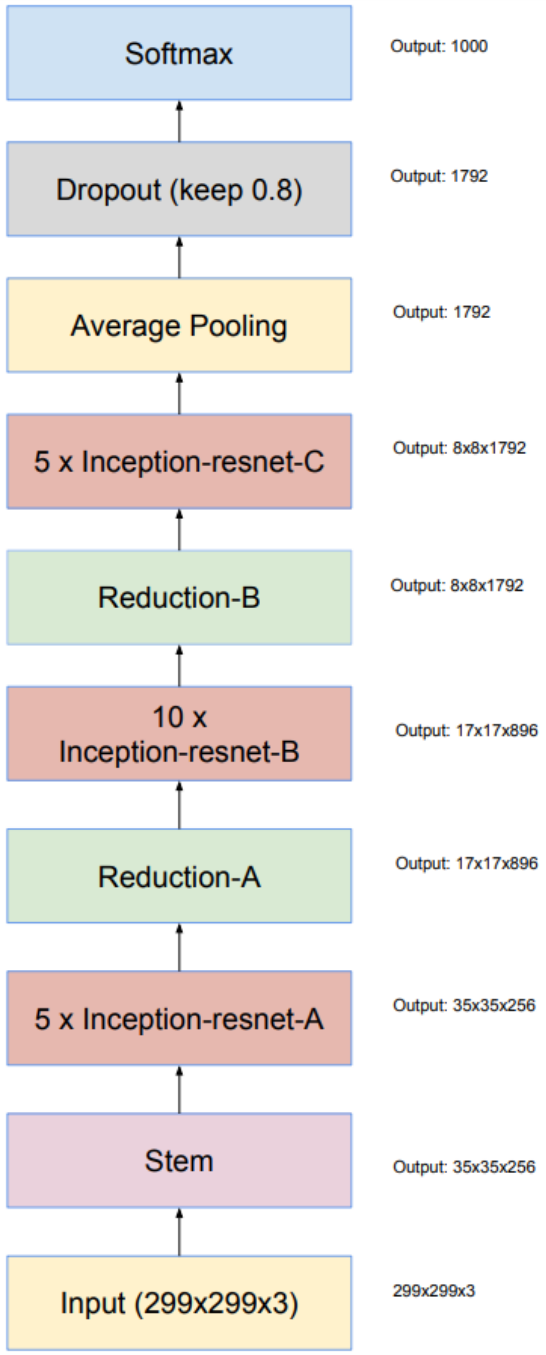
Each Inception block is followed by filter-expansion layer (1 × 1 convolution without activation) which is used for scaling up the dimensionality of the filter bank before the addition to match the depth of the input. This is needed to compensate for the dimensionality reduction induced by the Inception block.



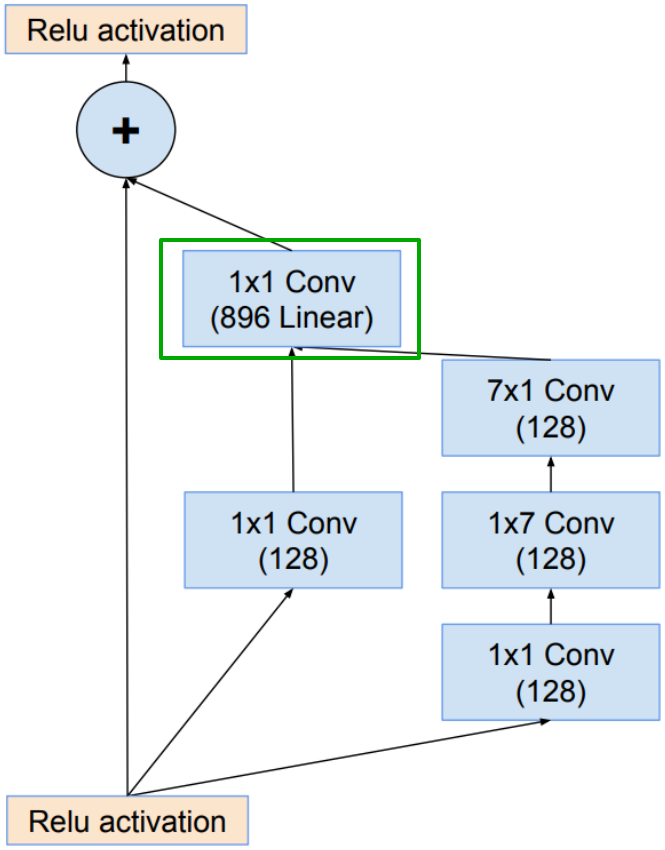


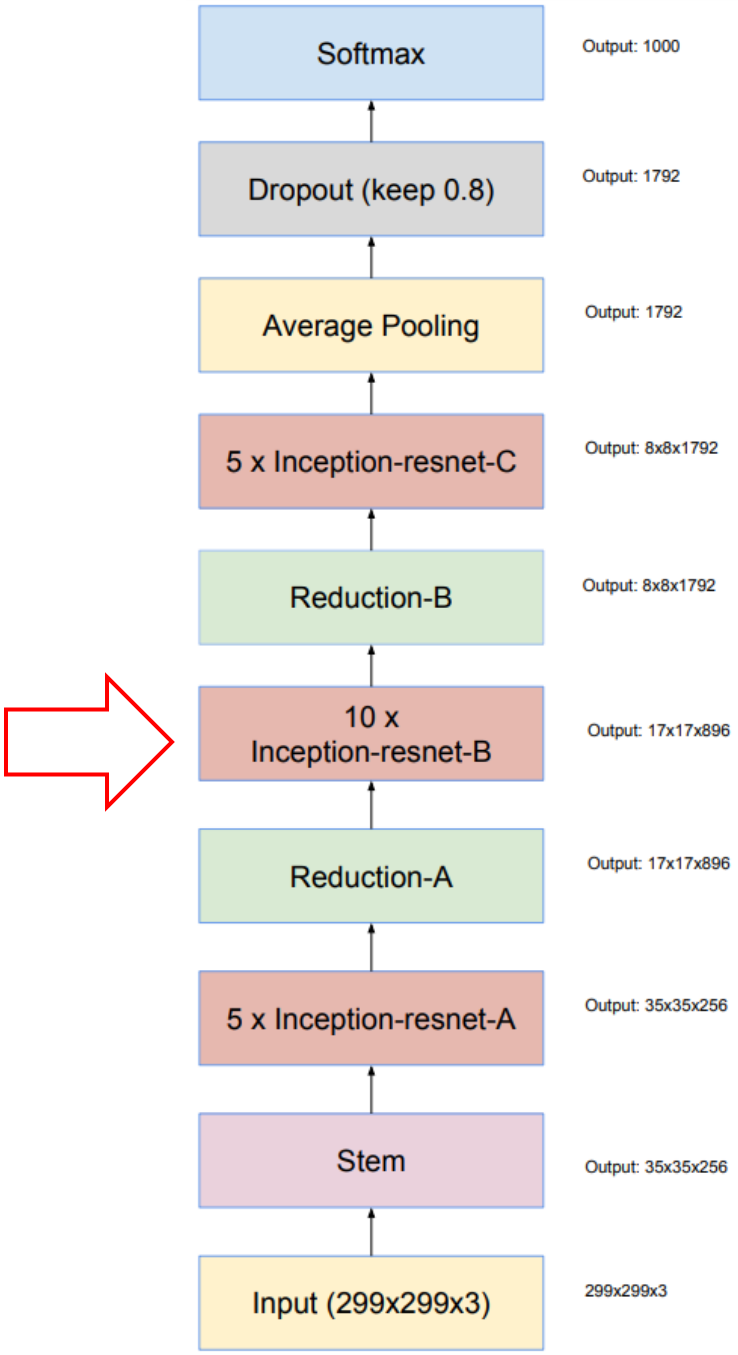
Inception-ResNet v1



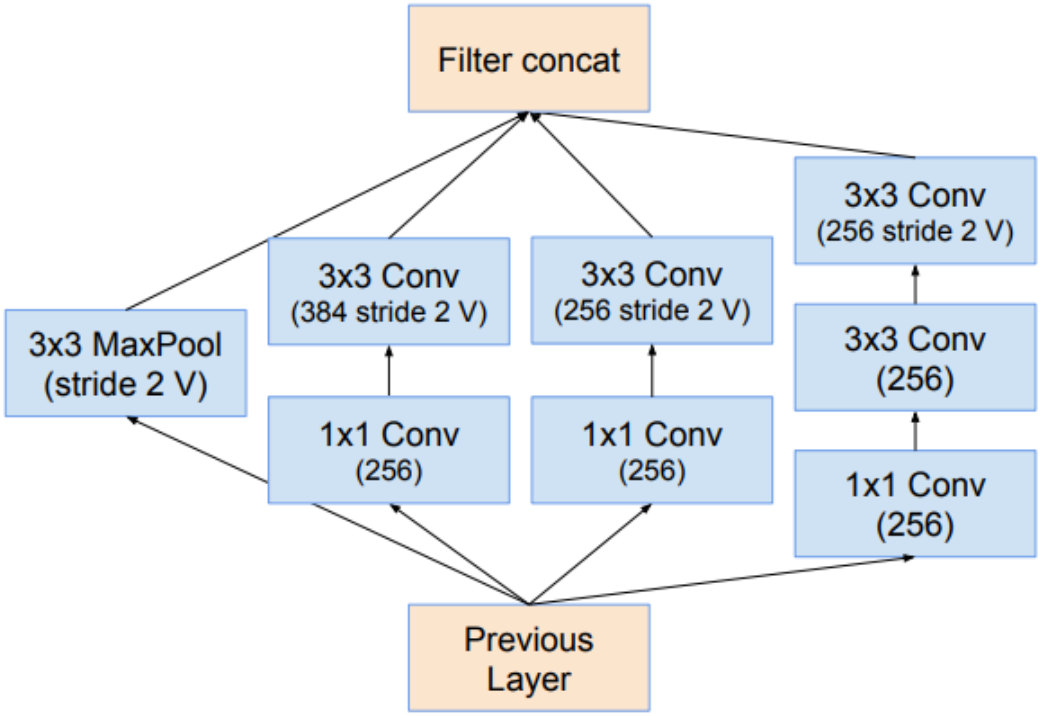


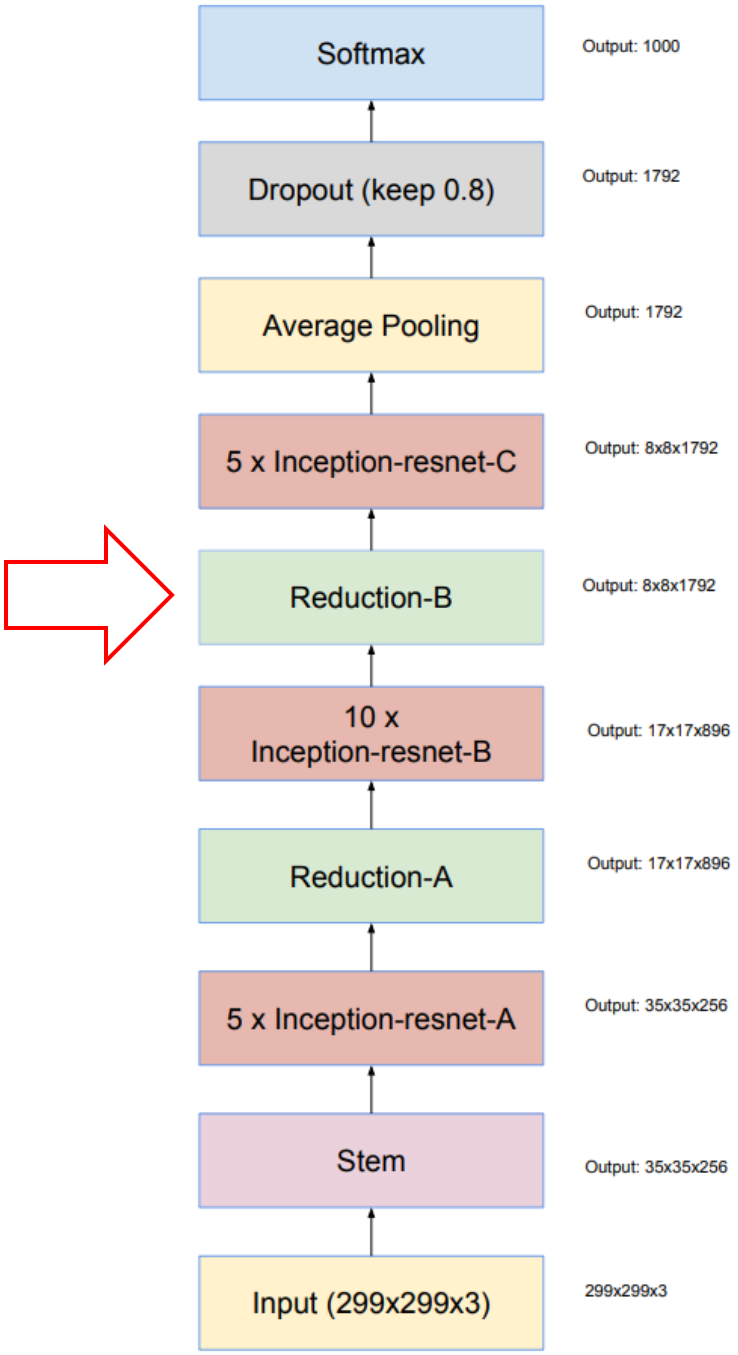
Inception-ResNet v1





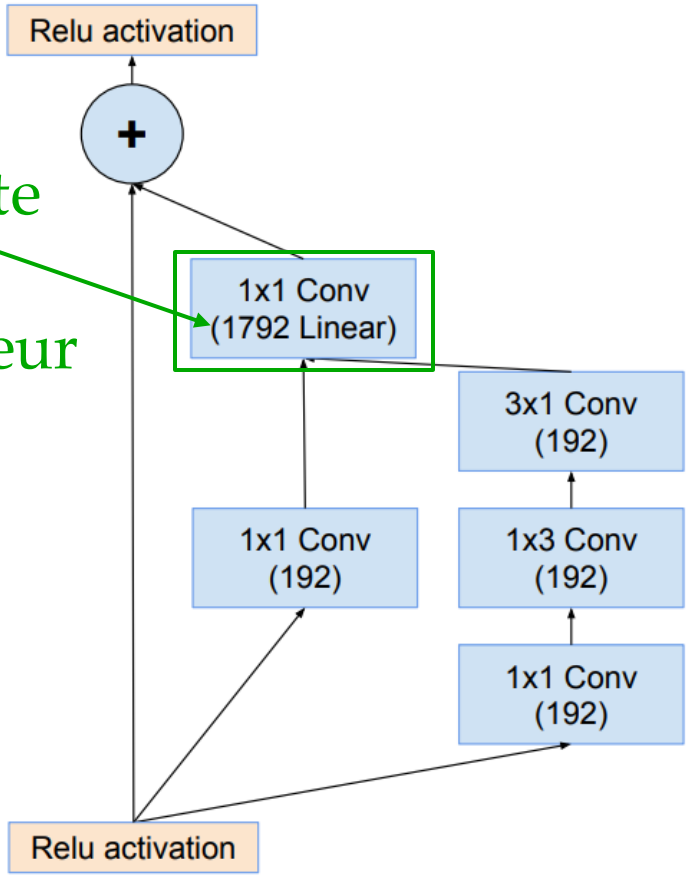
Inception-ResNet v1

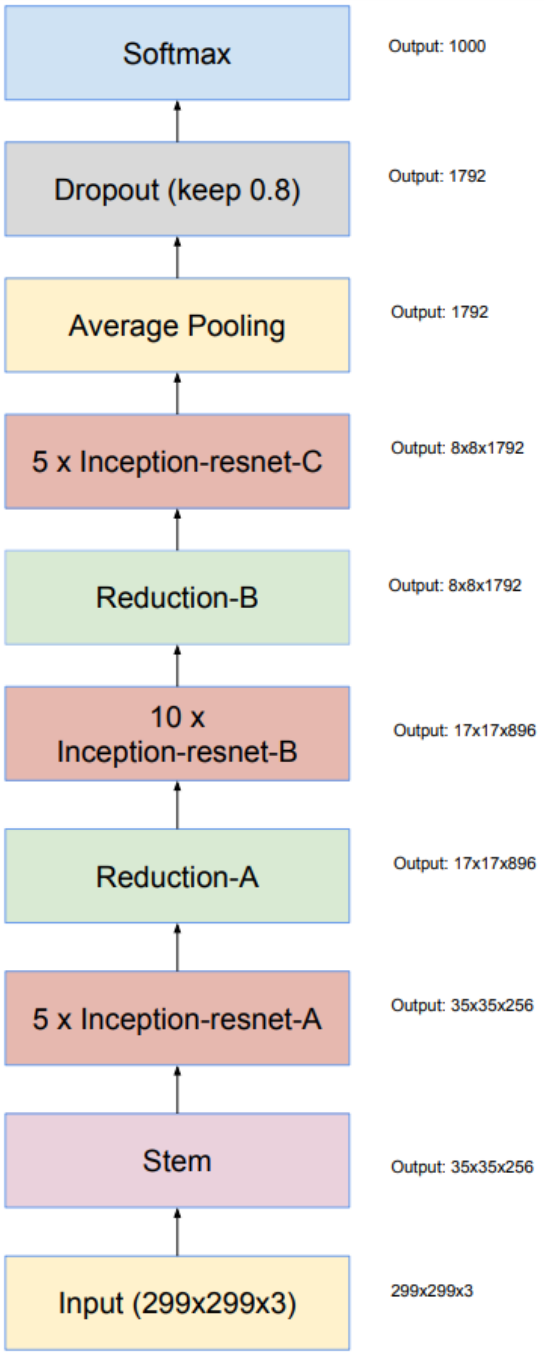




Inception-ResNet v1 et v2

augmente
avec la
profondeur





Résultats

- Connection résiduelle accélère l'entraînement
- Améliore les scores

Network	Crops	**Top-1 Error**	**Top-5 Error**
ResNet-151 [5]	dense	19.4%	4.5%
Inception-v3 [15]	144	18.9%	4.3%
Inception-ResNet-v1	144	18.8%	4.3%
Inception-v4	144	17.7%	3.8%
Inception-ResNet-v2	144	17.8%	3.7%

Table 4. 144 crops evaluations - single model experimental results. Reported on the all 50000 images of the validation set of ILSVRC 2012.

Ensembles

Network	Models	**Top-1 Error**	**Top-5 Error**
ResNet-151 [5]	6	–	3.6%
Inception-v3 [15]	4	17.3%	3.6%
Inception-v4 + 3× Inception-ResNet-v2	4	16.5%	3.1%

ResNeXt

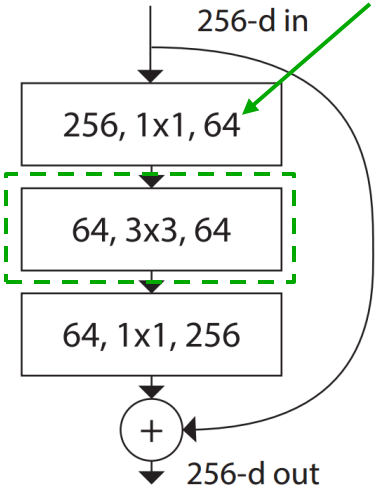
- Architecture multi-branche (comme Inception), mais répète la même topologie (contrairement à Inception)
- Le nombre de branche == *cardinality*
- Prétend qu'il est mieux d'augmenter la *cardinality* que la profondeur/largeur
- Cherche à améliorer la performance sous un budget fixe de FLOPS et de paramètres

S. Xie, R. Girshick, P. Dollar, Z. Tu and K. He. Aggregated Residual Transformations for Deep Neural Networks. arXiv preprint arXiv:1611.05431v1,2016.

ResNeXt

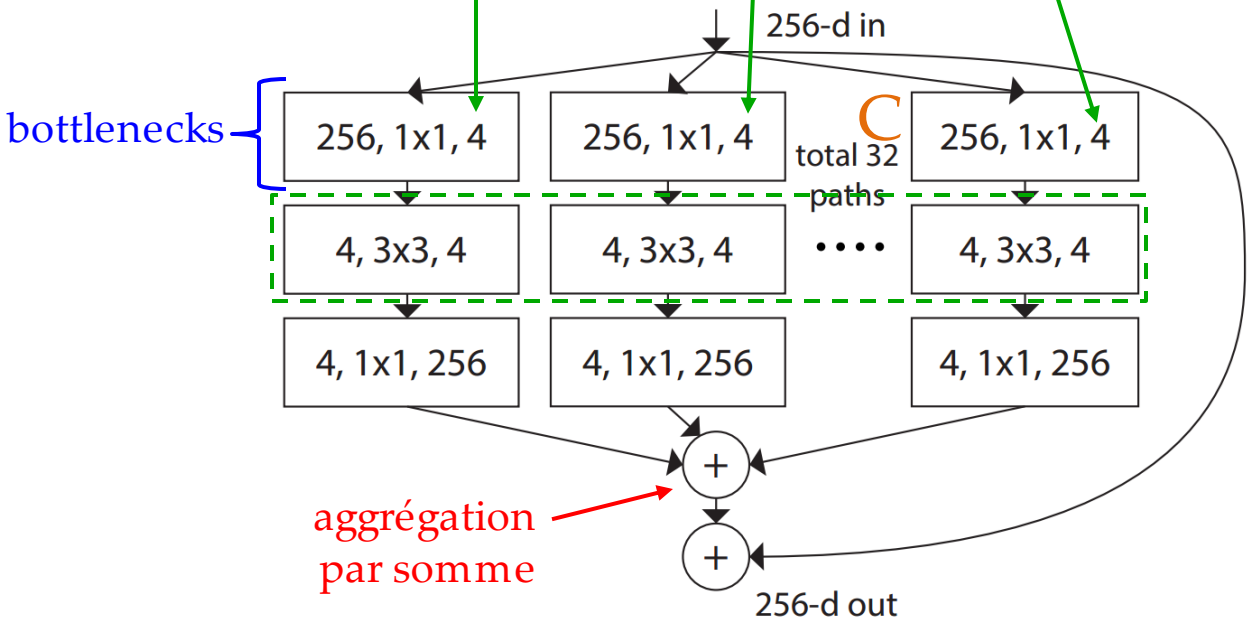
- Inception effectue *split-transform-merge*
- On peut voir Inception comme étant un sous-espace d'une couche ne contenant que des 5x5
- Le nombre de filtres 1x1, 3x3, 5x5 varie d'une couche à l'autre, difficile à tuner ☹
- ResNeXt va empruter d'inception la philosophie *split-transform-merge*

1 embedding de 64



embedding de basse dimension

plusieurs embedding de taille d, ici 4



cardinality C	1	2	4	8	32
width of bottleneck d	64	40	24	14	4
width of group conv.	64	80	96	112	128

Table 2. Relations between cardinality and width (for the template of conv2), with roughly preserved complexity on a residual block. The number of parameters is ~70k for the template of conv2. The number of FLOPs is ~0.22 billion (# params×56×56 for conv2).

ResNeXt

diminution/ augmentation au même rythme

stage	output	ResNet-50	**ResNeXt-50 (32×4d)**
conv1	112×112	7×7, 64, stride 2	7×7, 64, stride 2
conv2	56×56	3×3 max pool, stride 2 $\begin{bmatrix} 1\times1, 64 \\ 3\times3, 64 \\ 1\times1, 256 \end{bmatrix} \times 3$ répétitions	3×3 max pool, stride 2 $\begin{bmatrix} 1\times1, 128 \\ 3\times3, 128, C{=}32 \\ 1\times1, 256 \end{bmatrix} \times 3$
conv3	28×28	$\begin{bmatrix} 1\times1, 128 \\ 3\times3, 128 \\ 1\times1, 512 \end{bmatrix} \times 4$	$\begin{bmatrix} 1\times1, 256 \\ 3\times3, 256, C{=}32 \\ 1\times1, 512 \end{bmatrix} \times 4$
conv4	14×14	$\begin{bmatrix} 1\times1, 256 \\ 3\times3, 256 \\ 1\times1, 1024 \end{bmatrix} \times 6$	$\begin{bmatrix} 1\times1, 512 \\ 3\times3, 512, C{=}32 \\ 1\times1, 1024 \end{bmatrix} \times 6$
conv5	7×7	$\begin{bmatrix} 1\times1, 512 \\ 3\times3, 512 \\ 1\times1, 2048 \end{bmatrix} \times 3$	$\begin{bmatrix} 1\times1, 1024 \\ 3\times3, 1024, C{=}32 \\ 1\times1, 2048 \end{bmatrix} \times 3$
	1×1	global average pool 1000-d fc, softmax	global average pool 1000-d fc, softmax
# params.		$\mathbf{25.5}\times10^6$	$\mathbf{25.0}\times10^6$
FLOPs		$\mathbf{4.1}\times10^9$	$\mathbf{4.2}\times10^9$

ResNeXt : Résultats

cardinality

bottleneck

		setting	top-1 error (%)
capacité similaire	ResNet-50	1 × 64d	23.9
	ResNeXt-50	2 × 40d	23.0
	ResNeXt-50	4 × 24d	22.6
	ResNeXt-50	8 × 14d	22.3
	ResNeXt-50	32 × 4d	**22.2**
capacité similaire	ResNet-101	1 × 64d	22.0
	ResNeXt-101	2 × 40d	21.7
	ResNeXt-101	4 × 24d	21.4
	ResNeXt-101	8 × 14d	21.3
	ResNeXt-101	32 × 4d	**21.2**

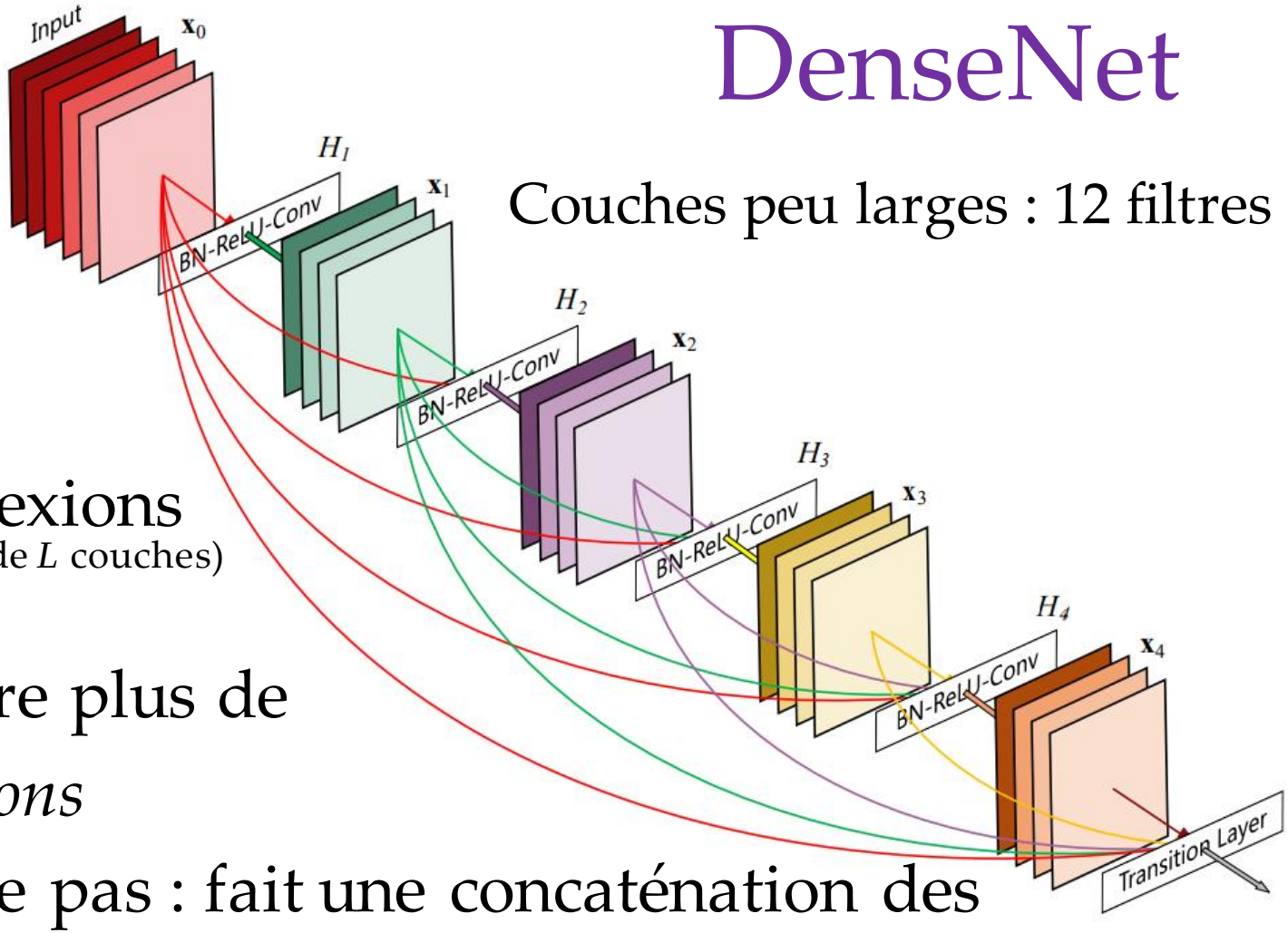
Table 3. Ablation experiments on ImageNet-1K. (**Top**): ResNet-50 with preserved complexity (∼4.1 billion FLOPs); (**Bottom**): ResNet-101 with preserved complexity (∼7.8 billion FLOPs). The error rate is evaluated on the single crop of 224×224 pixels.

Si on double la capacité

	setting	top-1 err (%)	top-5 err (%)
1× complexity references:			
ResNet-101	1 × 64d	22.0	6.0
ResNeXt-101	32 × 4d	21.2	5.6
2× complexity models follow:			
ResNet-**200** [15]	1 × 64d	21.7	5.8
ResNet-101, wider	1 × **100**d	21.3	5.7
ResNeXt-101	**2** × 64d	20.7	5.5
ResNeXt-101	**64** × 4d	**20.4**	**5.3**

Table 4. Comparisons on ImageNet-1K when the number of FLOPs is increased to 2× of ResNet-101's. The error rate is evaluated on the single crop of 224×224 pixels. The highlighted factors are the factors that increase complexity.

DenseNet



Couches peu larges : 12 filtres

$\frac{L(L+1)}{2}$ connexions (par bloc de L couches)

- Ajoute encore plus de *skip connections*
- N'additionne pas : fait une concaténation des *features* des couches précédentes

ResNet : $\mathbf{x}_\ell = H_\ell(\mathbf{x}_{\ell-1}) + \mathbf{x}_{\ell-1}$

DenseNet : $\mathbf{x}_\ell = H_\ell([\mathbf{x}_0, \mathbf{x}_1, \ldots, \mathbf{x}_{\ell-1}])$

concaténation

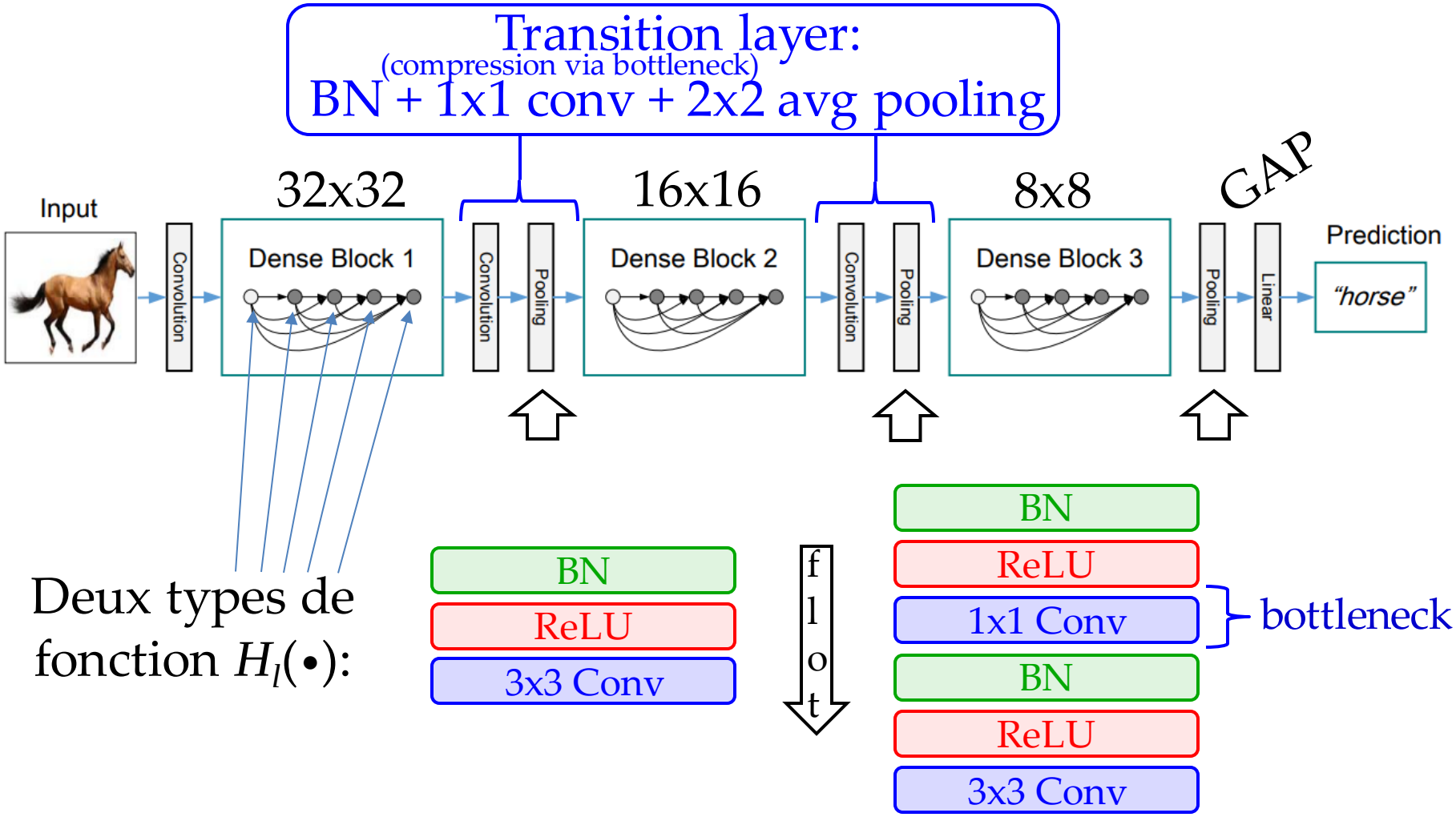
Huang et al., Densely Connected Convolutional Networks, 2016.

DenseNet

- Diminue *vanishing gradient*
- Augmente la propagation et réutilisation de *features*
- Semble régulariser
- Réduit le nombre de paramètres :
  - Conv standard : doit apprendre implicitement quels *features* laisser passer
  - ResNet : explicite, mais certaines couches contribuent peu (voir Stoch. Depth), donc des poids inutiles

DenseNet

- Connection vers toutes les couches subséquentes qui ont la même taille de feature map (Dense Block)



DenseNet (ImageNet)

Layers	Output Size	DenseNet-121	DenseNet-169	DenseNet-201	DenseNet-264
Convolution	112 × 112	7 × 7 conv, stride 2			
Pooling	56 × 56	3 × 3 max pool, stride 2			
Dense Block (1)	56 × 56	$\begin{bmatrix} 1 \times 1 \text{ conv} \\ 3 \times 3 \text{ conv} \end{bmatrix} \times 6$	$\begin{bmatrix} 1 \times 1 \text{ conv} \\ 3 \times 3 \text{ conv} \end{bmatrix} \times 6$	$\begin{bmatrix} 1 \times 1 \text{ conv} \\ 3 \times 3 \text{ conv} \end{bmatrix} \times 6$	$\begin{bmatrix} 1 \times 1 \text{ conv} \\ 3 \times 3 \text{ conv} \end{bmatrix} \times 6$
Transition Layer (1)	56 × 56	1 × 1 conv			
	28 × 28	2 × 2 average pool, stride 2			
Dense Block (2)	28 × 28	$\begin{bmatrix} 1 \times 1 \text{ conv} \\ 3 \times 3 \text{ conv} \end{bmatrix} \times 12$	$\begin{bmatrix} 1 \times 1 \text{ conv} \\ 3 \times 3 \text{ conv} \end{bmatrix} \times 12$	$\begin{bmatrix} 1 \times 1 \text{ conv} \\ 3 \times 3 \text{ conv} \end{bmatrix} \times 12$	$\begin{bmatrix} 1 \times 1 \text{ conv} \\ 3 \times 3 \text{ conv} \end{bmatrix} \times 12$
Transition Layer (2)	28 × 28	1 × 1 conv			
	14 × 14	2 × 2 average pool, stride 2			
Dense Block (3)	14 × 14	$\begin{bmatrix} 1 \times 1 \text{ conv} \\ 3 \times 3 \text{ conv} \end{bmatrix} \times 24$	$\begin{bmatrix} 1 \times 1 \text{ conv} \\ 3 \times 3 \text{ conv} \end{bmatrix} \times 32$	$\begin{bmatrix} 1 \times 1 \text{ conv} \\ 3 \times 3 \text{ conv} \end{bmatrix} \times 48$	$\begin{bmatrix} 1 \times 1 \text{ conv} \\ 3 \times 3 \text{ conv} \end{bmatrix} \times 64$
Transition Layer (3)	14 × 14	1 × 1 conv			
	7 × 7	2 × 2 average pool, stride 2			
Dense Block (4)	7 × 7	$\begin{bmatrix} 1 \times 1 \text{ conv} \\ 3 \times 3 \text{ conv} \end{bmatrix} \times 16$	$\begin{bmatrix} 1 \times 1 \text{ conv} \\ 3 \times 3 \text{ conv} \end{bmatrix} \times 32$	$\begin{bmatrix} 1 \times 1 \text{ conv} \\ 3 \times 3 \text{ conv} \end{bmatrix} \times 32$	$\begin{bmatrix} 1 \times 1 \text{ conv} \\ 3 \times 3 \text{ conv} \end{bmatrix} \times 48$
Classification Layer	1 × 1	7 × 7 global average pool			
		1000D fully-connected, softmax			

DenseNet

Sans data augm. (effet régularisateur)

Method	Depth	Params	C10	C10+	C100	C100+	SVHN
Network in Network [22]	-	-	10.41	8.81	35.68	-	2.35
All-CNN [32]	-	-	9.08	7.25	-	33.71	-
Deeply Supervised Net [20]	-	-	9.69	7.97	-	34.57	1.92
Highway Network [34]	-	-	-	7.72	-	32.39	-
FractalNet [17]	21	38.6M	10.18	5.22	35.34	23.30	2.01
with Dropout/Drop-path	21	38.6M	7.33	4.60	28.20	23.73	1.87
ResNet [11]	110	1.7M	-	6.61	-	-	-
ResNet (reported by [13])	110	1.7M	13.63	6.41	44.74	27.22	2.01
ResNet with Stochastic Depth [13]	110	1.7M	11.66	5.23	37.80	24.58	1.75
	1202	10.2M	-	4.91	-	-	-
Wide ResNet [42]	16	11.0M	-	4.81	-	22.07	-
	28	36.5M	-	4.17	-	20.50	-
with Dropout	16	2.7M	-	-	-	-	1.64
ResNet (pre-activation) [12]	164	1.7M	11.26*	5.46	35.58*	24.33	-
	1001	10.2M	10.56*	4.62	33.47*	22.71	-
DenseNet ($k = 12$)	40	1.0M	**7.00**	5.24	**27.55**	24.42	1.79
DenseNet ($k = 12$)	100	7.0M	**5.77**	**4.10**	**23.79**	**20.20**	1.67
DenseNet ($k = 24$)	100	27.2M	**5.83**	**3.74**	**23.42**	**19.25**	**1.59**
DenseNet-BC ($k = 12$)	100	0.8M	**5.92**	4.51	**24.15**	22.27	1.76
DenseNet-BC ($k = 24$)	250	15.3M	**5.19**	**3.62**	**19.64**	**17.60**	1.74
DenseNet-BC ($k = 40$)	190	25.6M	-	**3.46**	-	**17.18**	-

taille similaire

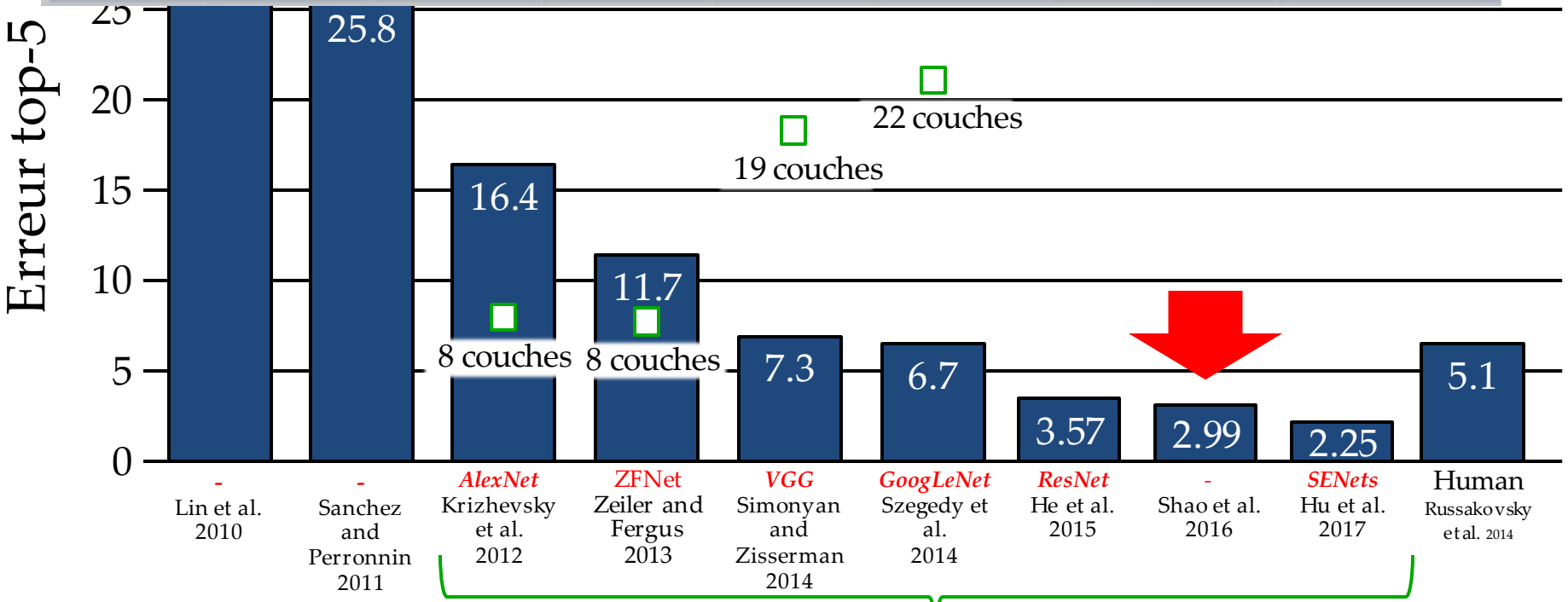
taille similaire

Table 2: Error rates (%) on CIFAR and SVHN datasets. k denotes network's growth rate. Results that surpass all competing methods are **bold** and the overall best results are **blue**. "+" indicates standard data augmentation (translation and/or mirroring). $*$ indicates results run by ourselves. All the results of DenseNets without data augmentation (C10, C100, SVHN) are obtained using Dropout. DenseNets achieve lower error rates while using fewer parameters than ResNet. Without data augmentation, DenseNet performs better by a large margin.

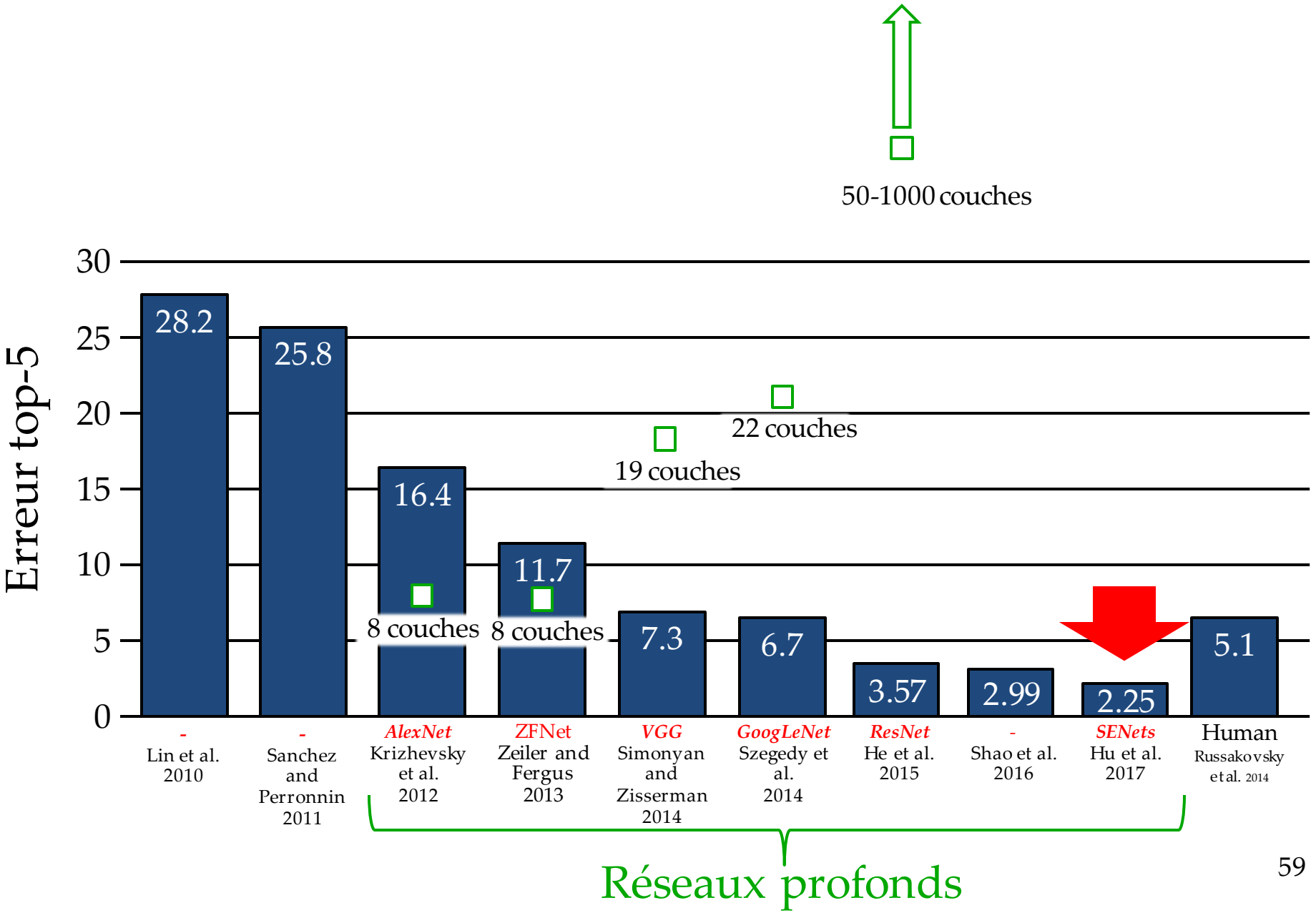
Large Scale Visual Recognition Challenge

Shao et al. : ensemble, peu intéressant

	Inception-v3	Inception-v4	Inception-Resnet-v2	Resnet-200	Wrn-68-3	Fusion (Val.)	Fusion (Test)
Err. (%)	4.20	4.01	3.52	4.26	4.65	2.92 (-0.6)	2.99



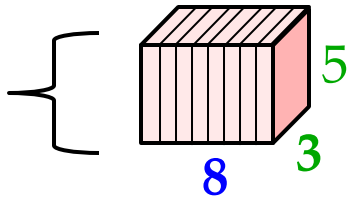
Large Scale Visual Recognition Challenge



Squeeze and Excite Net : SENet

- Exemple de **microarchitecture**
- convnet standard entremêle l'information **spatiale** et du domaine des features

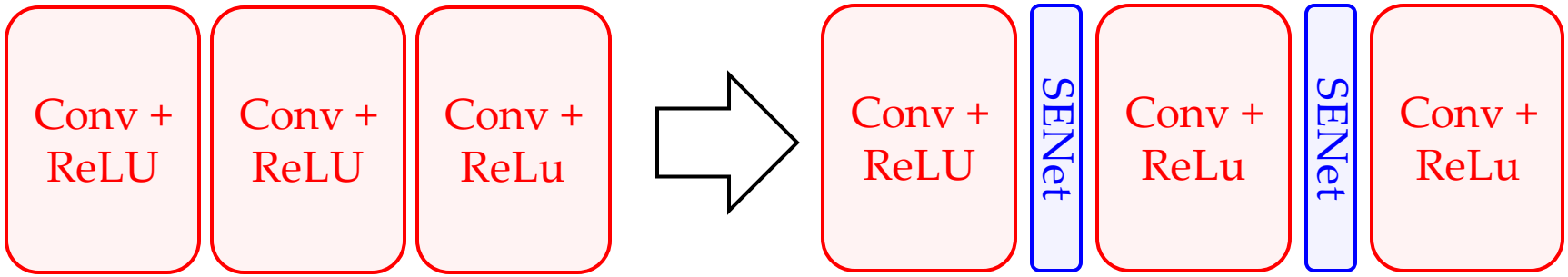
Dans une région spatiale donnée (localité)



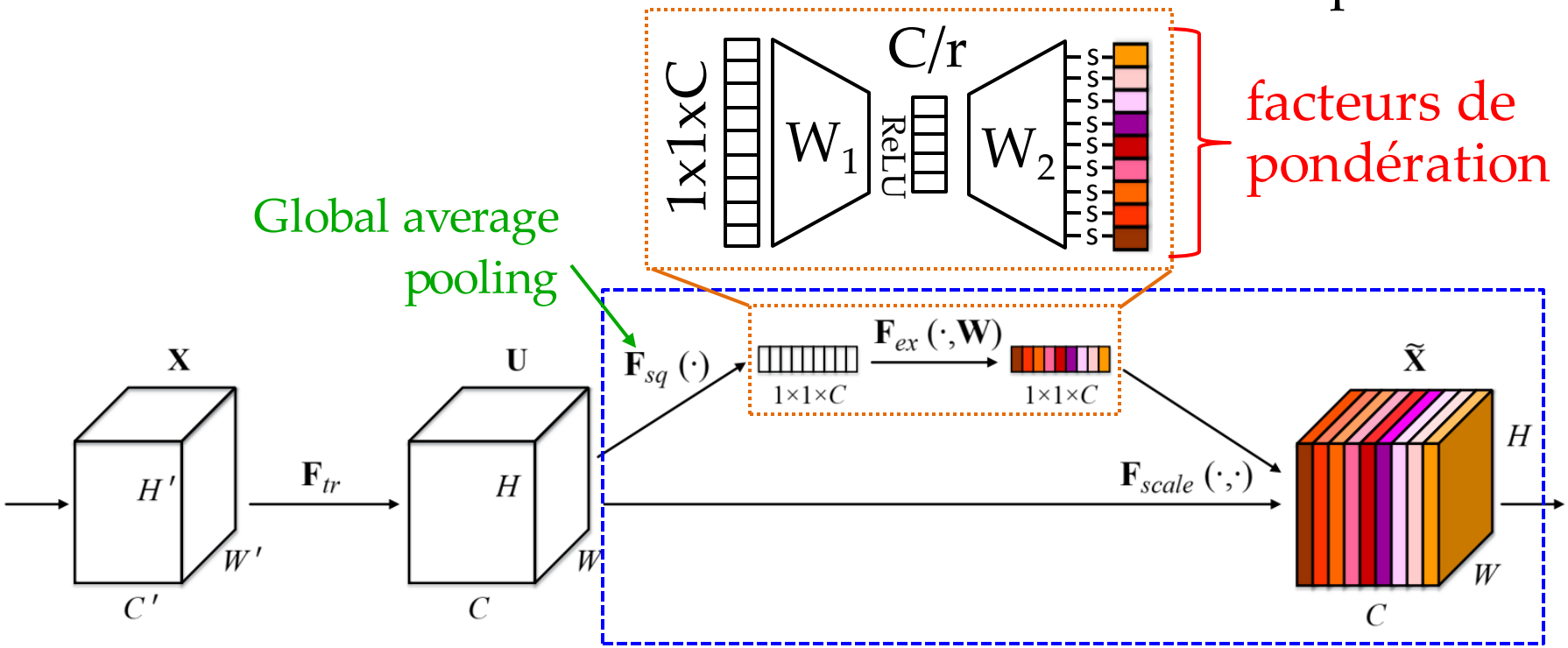
- SENet: pondérer les *feature maps* **globalement** avec un micro-réseau
- Ajoute très peu de paramètres et de calcul
  - architectures sont déjà larges
- Idée du *bottleneck* qui revient

SENet

- Ajout tel-que-tel (*drop-in*)



r : ratio de compression

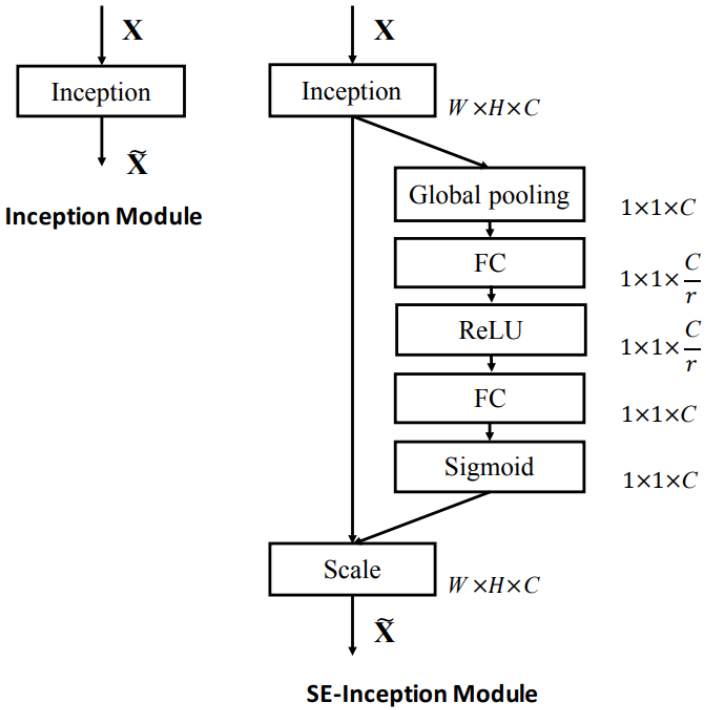


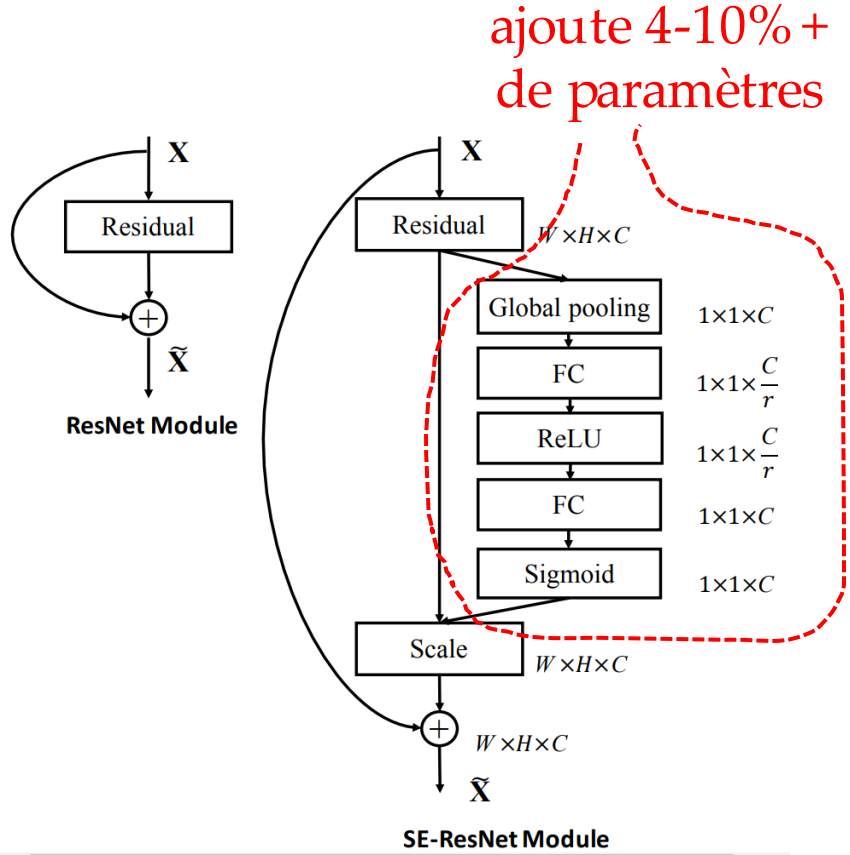
SENet

- Modéliser les interdépendances entre *feature maps* avec un micro-réseau
- Appelle ce mécanisme *feature recalibration*
- Forme d'**attention**, mais selon les *feature map*
- Utilise information globale pour rehausser des *features* utiles et réprimer les moins utiles

SENet

- Testés sur différentes architectures
  - ResNet
  - Inception et Inception-ResNet
  - ResNeXt





SENet : Résultats ImageNet

Différence négligeable en calcul

	original		re-implementation			SENet		
	top-1 err.	top-5 err.	top-1 err.	top-5 err.	GFLOPs	top-1 err.	top-5 err.	GFLOPs
ResNet-50 [9]	24.7	7.8	24.80	7.48	3.86	$23.29_{(1.51)}$	$6.62_{(0.86)}$	3.87
ResNet-101 [9]	23.6	7.1	23.17	6.52	7.58	$22.38_{(0.79)}$	$6.07_{(0.45)}$	7.60
ResNet-152 [9]	23.0	6.7	22.42	6.34	11.30	$21.57_{(0.85)}$	$5.73_{(0.61)}$	11.32
ResNeXt-50 [43]	22.2	-	22.11	5.90	4.24	$21.10_{(1.01)}$	$5.49_{(0.41)}$	4.25
ResNeXt-101 [43]	21.2	5.6	21.18	5.57	7.99	$20.70_{(0.48)}$	$5.01_{(0.56)}$	8.00
BN-Inception [14]	25.2	7.82	25.38	7.89	2.03	$24.23_{(1.15)}$	$7.14_{(0.75)}$	2.04
Inception-ResNet-v2 [38]	$19.9^{\dagger}$	$4.9^{\dagger}$	20.37	5.21	11.75	$19.80_{(0.57)}$	$4.79_{(0.42)}$	11.76

Gains partout

valeur utilisée ⇩

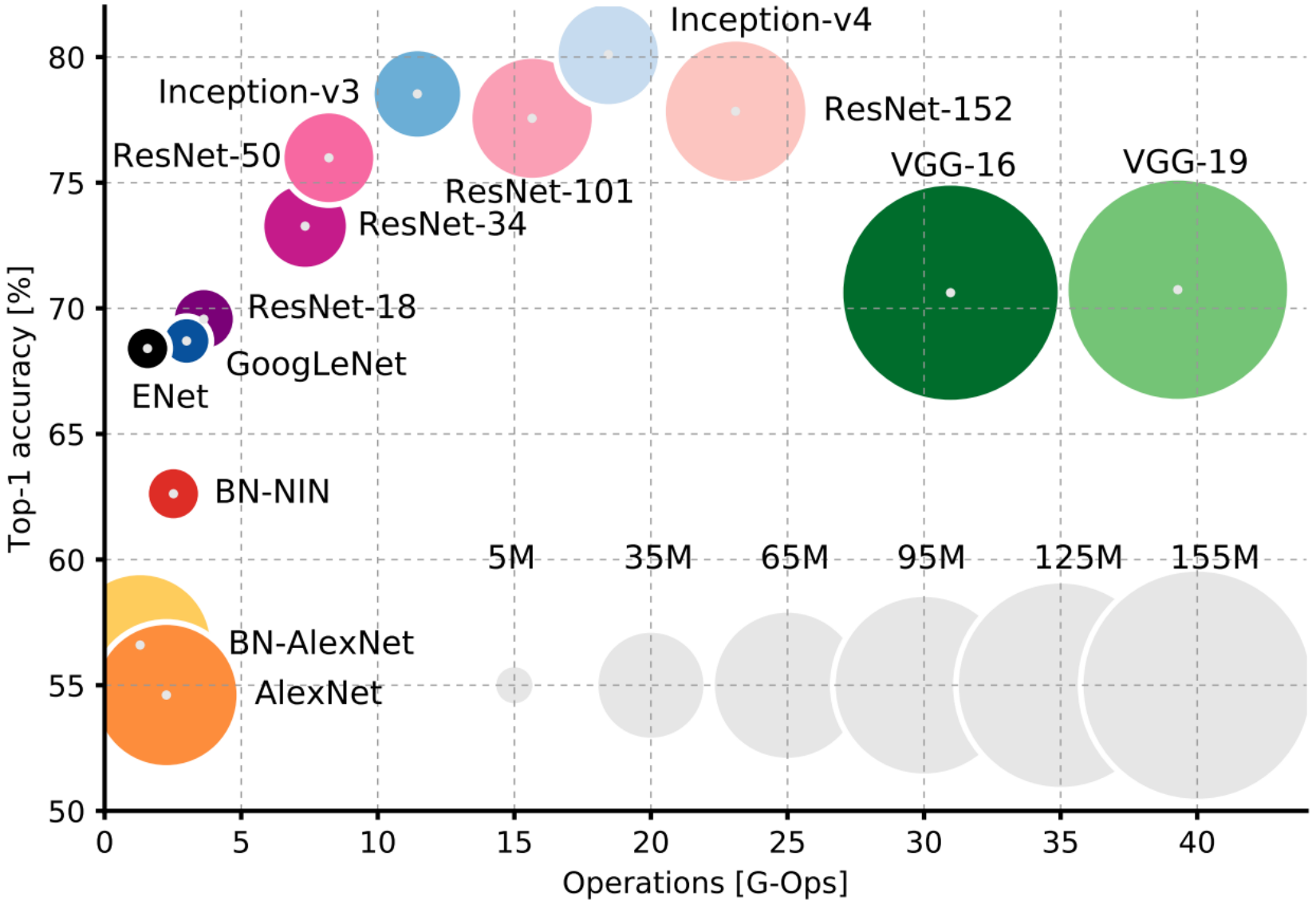
Ratio r	top-1 err.	top-5 err.	model size (MB)
4	23.21	6.63	137
8	23.19	6.64	117
16	23.29	6.62	108
32	23.40	6.77	103
original	24.80	7.48	98

Facteur de réduction de dimensionnalité

SENet

- Rôle joué semble varier en fonction de la profondeur
  - en bas, les excitations des *features* sont peu corrélées avec la classe
  - en haut, les excitations des *features* sont plus spécialisés (en fonction de la classe)

Efficacité des réseaux

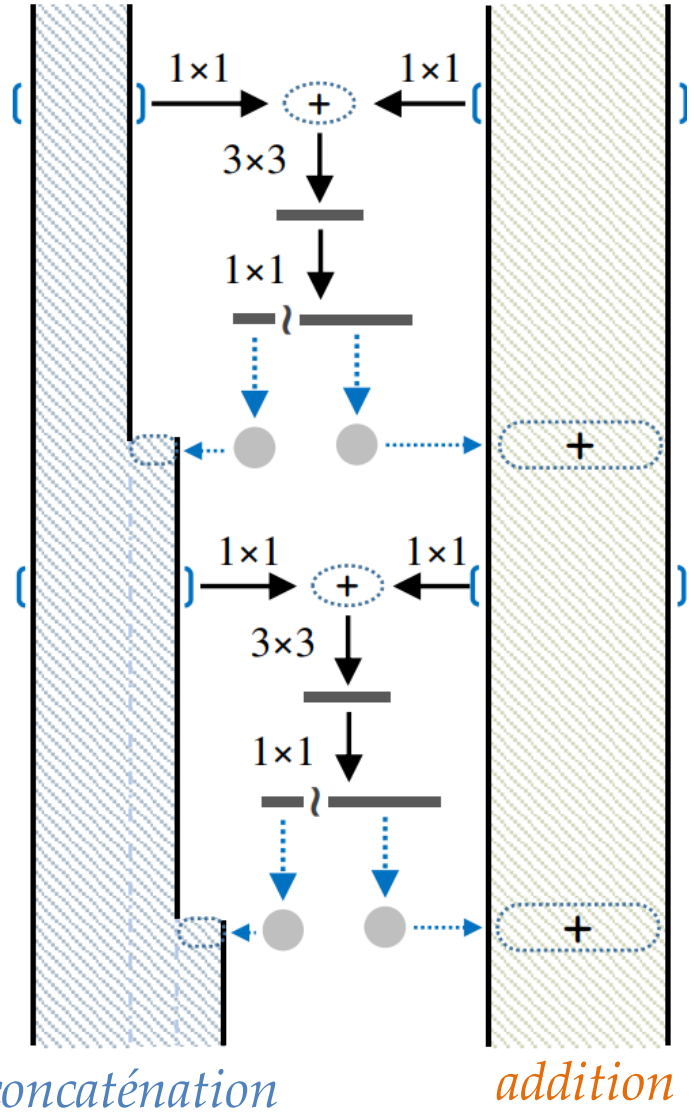


A. Canziani et al. An Analysis of Deep Neural Network Models for Practical Applications, 2017.

Autres architectures

Dual Path network

- À chaque bloc, devrait-on :
  - additionner, ou
  - concaténer
- ResNet : additionne
- Inception : concatène
- DenseNet : concatène
- Dual Path : fait les deux!



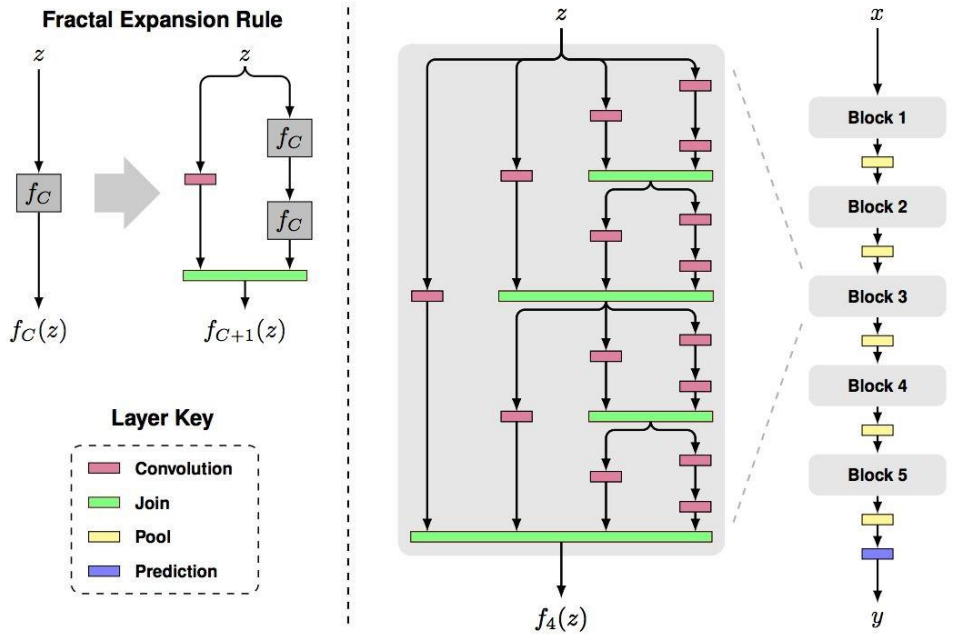
https://arxiv.org/abs/1707.01629

FractalNet

FractalNet: Ultra-Deep Neural Networks without Residuals

[Larsson et al. 2017]

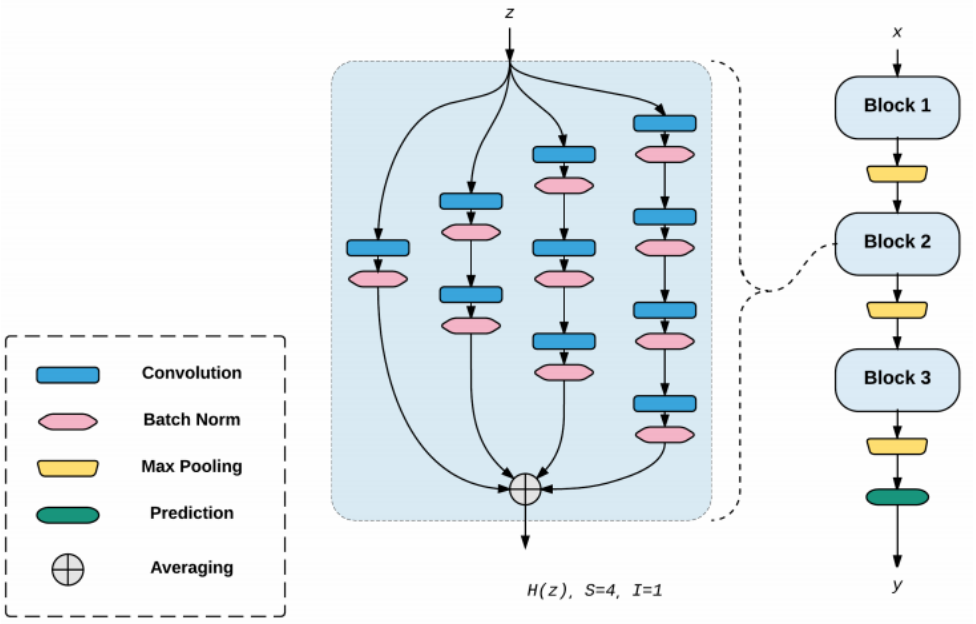
- Argues that key is transitioning effectively from shallow to deep and residual representations are not necessary
- Fractal architecture with both shallow and deep paths to output
- Trained with dropping out sub-paths
- Full network at test time



Figures copyright Larsson et al., 2017. Reproduced with permission.

tiré de : cs231n, Université Stanford

CresendoNet



CRESCENDONET: A SIMPLE DEEP CONVOLUTIONAL NEURAL NETWORK WITH ENSEMBLE BEHAVIOR, https://arxiv.org/pdf/1710.11176.pdf

CresendoNet

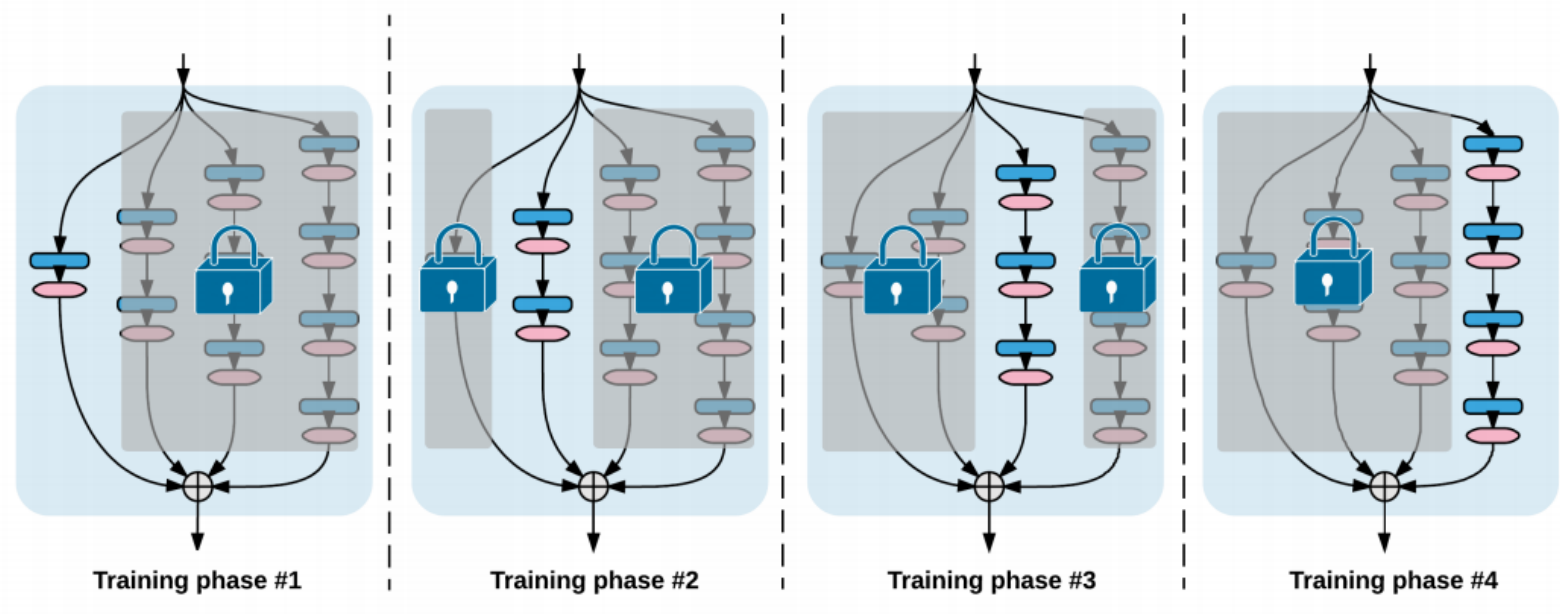


Figure 2: Path-wise training procedure.

Drop-path

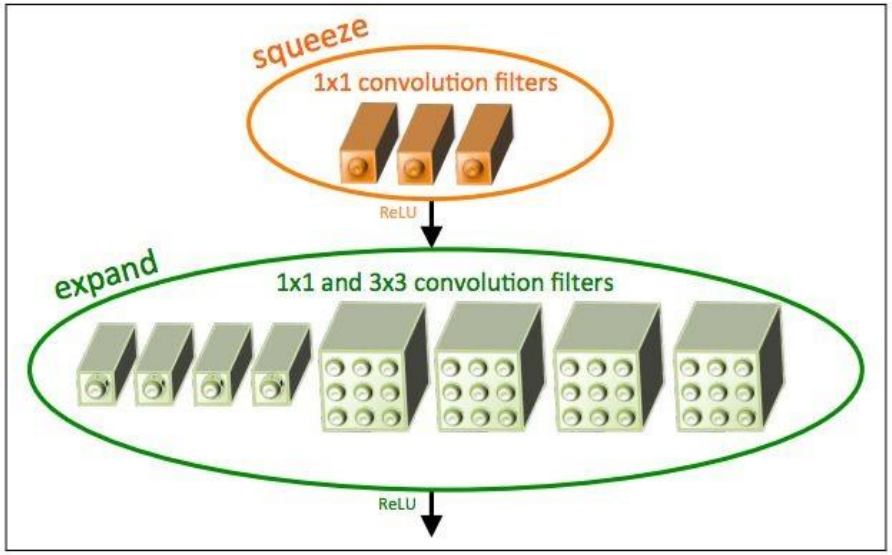
CRESCENDONET: A SIMPLE DEEP CONVOLUTIONAL NEURAL NETWORK WITH ENSEMBLE BEHAVIOR, https://arxiv.org/pdf/1710.11176.pdf

SqueezeNet

SqueezeNet: AlexNet-level Accuracy With 50x Fewer Parameters and <0.5Mb Model Size

[Iandola et al. 2017]

- Fire modules consisting of a 'squeeze' layer with 1x1 filters feeding an 'expand' layer with 1x1 and 3x3 filters
- AlexNet level accuracy on ImageNet with 50x fewer parameters
- Can compress to 510x smaller than AlexNet (0.5Mb)



Fin!