

Invatare automata in vedere artificiala

Curs 3

Convolutii

Retele Convolutionale (CNNs)

Batch Normalization

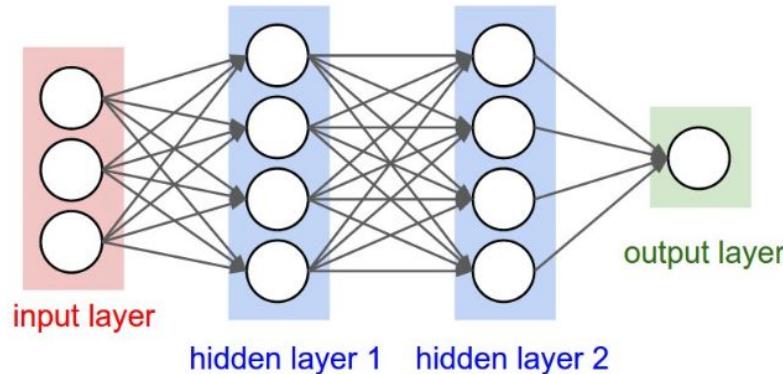
Structura cursului

- **Convolutii**
- **ImageNet challenge (ILSVRC)**
 - LeNet (1998)
 - AlexNet (2012)
 - VGGNet (2014)
 - GoogLeNet (2014)
 - ResNet (2015)
 - Inception-ResNet (2016)
 - NASNet (2017)
 - MobileNets (2017) - real-time inference on mobile devices
- Nice Reading: [CNN architectures](#)

Convoluții

Rețele Neurale Fully-Connected

- Input 1D (un vector), care trece printr-o serie de transformări (hidden layers);
- Fiecare strat este alcătuit din neuroni;
- Fiecare neuron este conectat la toți neuronii din stratul anterior;
- Neuronii același strat nu sunt conectați între ei;
- Ultimul strat se numește output. Probabilitati în problemele de clasificare.

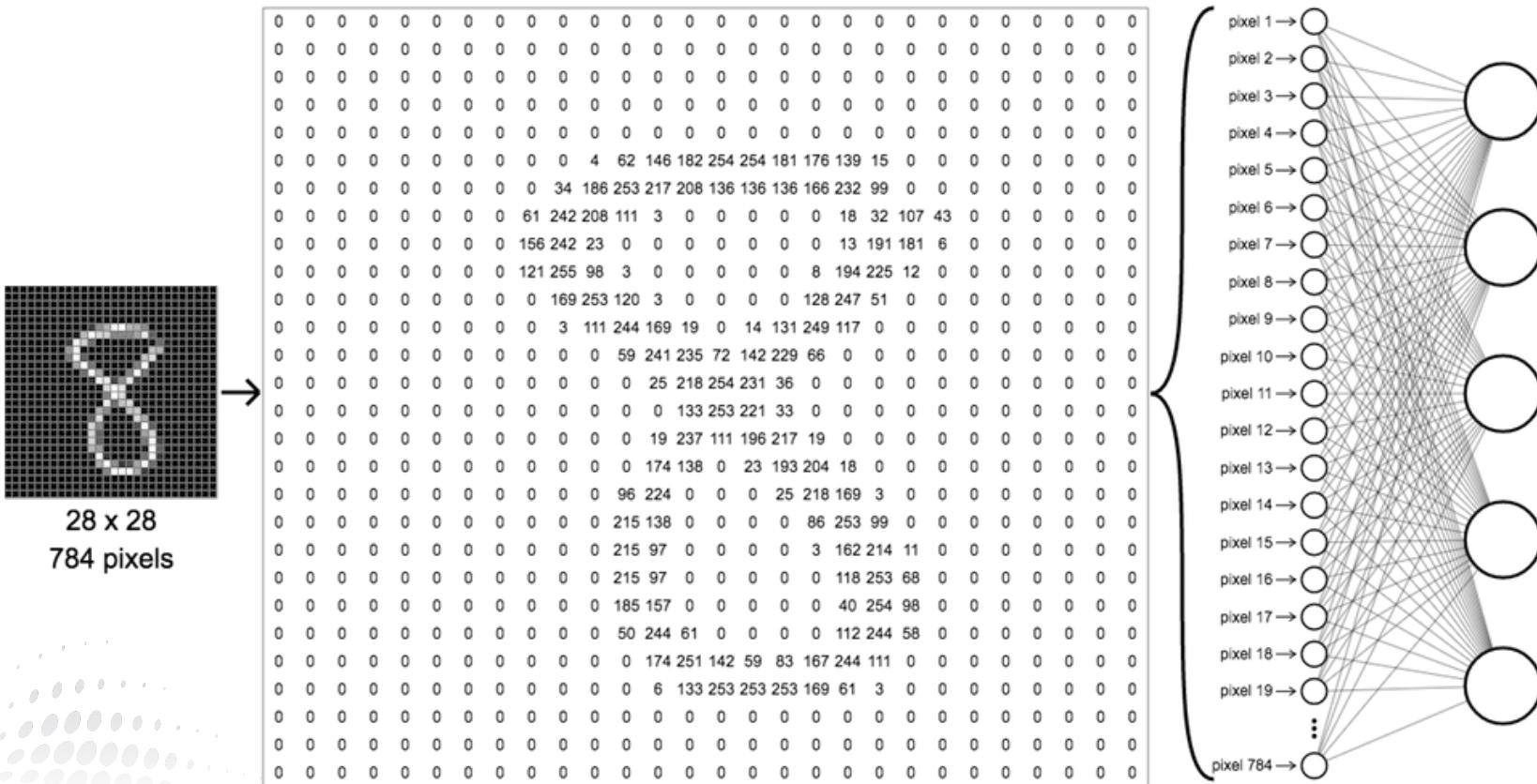


Rețele Neurale Fully-Connected

- Probleme:
 - Nu scalează cu dimensiunea imaginilor
 - e.g.: În MNIST, dimensiunea imaginilor este de doar 28x28x1 (grayscale). Astfel, un singur neuron de pe primul strat ascuns al rețelei are $28 \times 28 \times 1 = 784$ parametri. Este posibil, dar nu este scalabil pentru imagini mai mari
 - pentru o imagine de dimensiune 224x224x3 (rgb), neuronii ar avea 150,528 parametri, putând duce rapid la overfitting
 - Nu sunt invariante la translație: reacționează diferit față de o imagine și versiunea translatată a acesteia
 - Se pierde informația spațială când imaginile sunt vectorizate pentru rețea

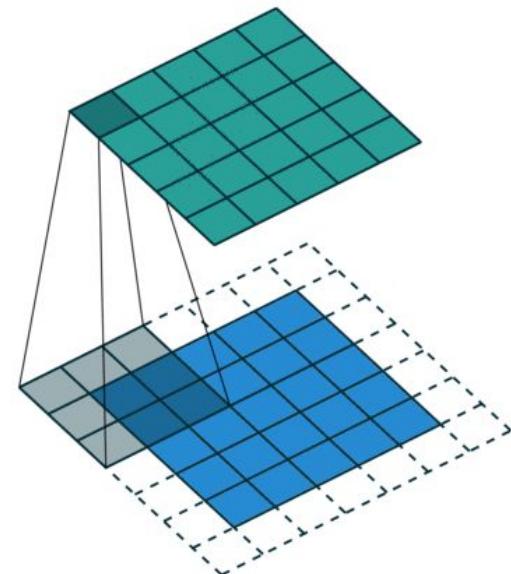


Retele Neurale Fully-Connected



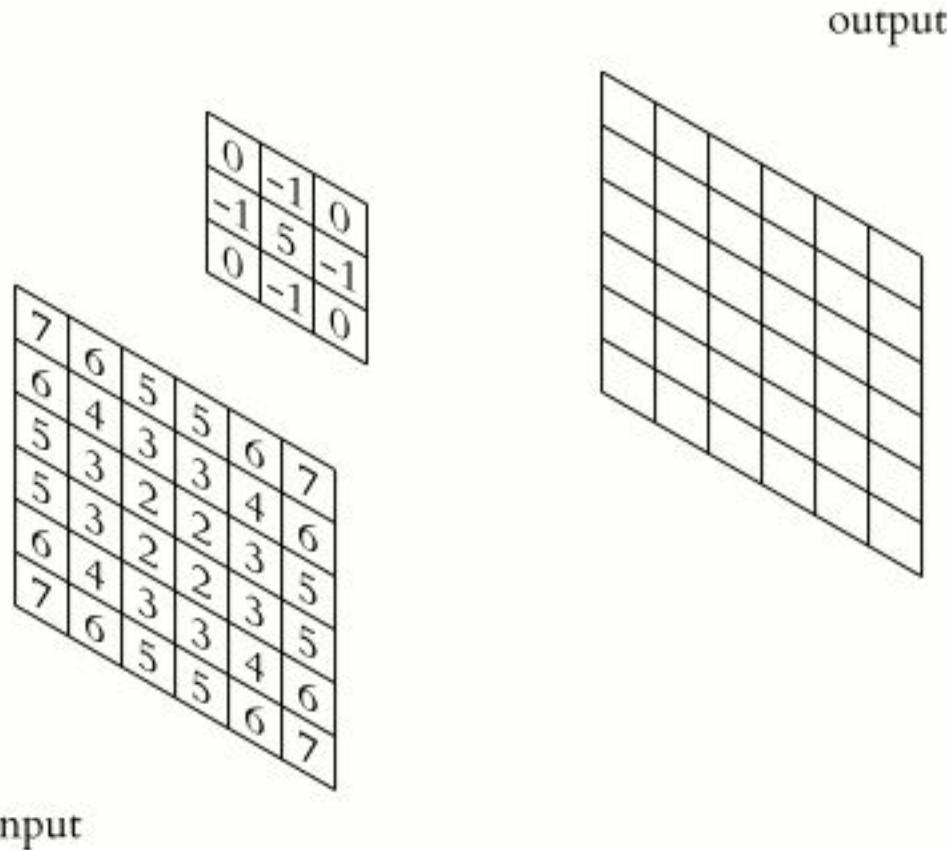
Convoluție

- Filtru / kernel cu parametri învățabili
- Fiecare filtru este mic pe înălțime și lățime (dimensiuni standard: 1x1, 3x3, 5x5, 7x7), dar se extinde pe toată adâncimea volumului de intrare
- În timpul antrenării, se glisează fiecare filtru de-a lungul lățimii și înălțimii volumului anterior și se calculează produsul scalar între elementele sale și regiunea corespunzătoare din input, la fiecare poziție
- Rezultatul conoluției este o hartă 2D (activation map), reprezentând răspunsul aplicării filtrului respectiv la fiecare poziție din input



http://deeplearning.net/software/theano/_images/same_padding_no_strides.gif

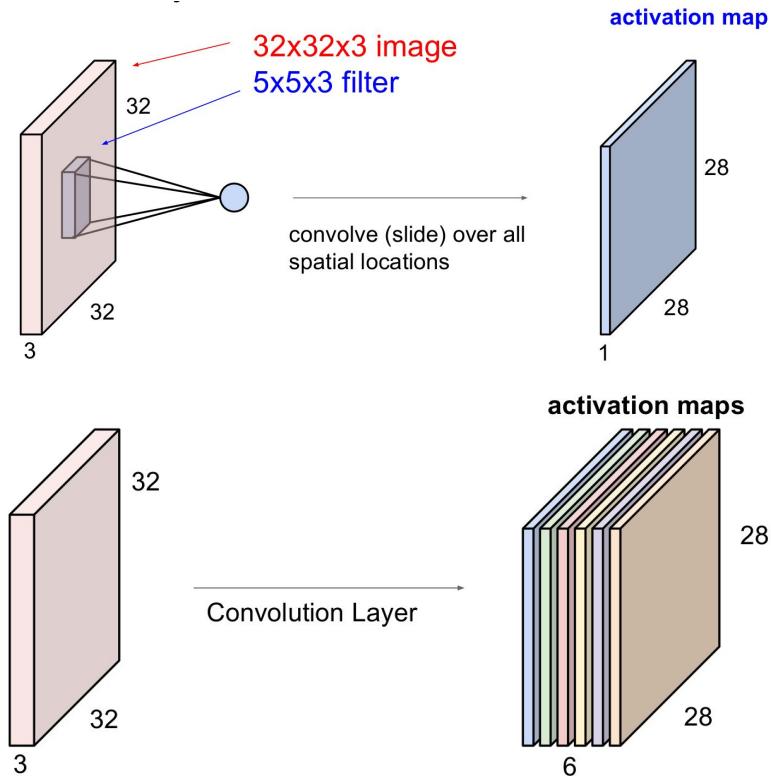
Convoluție



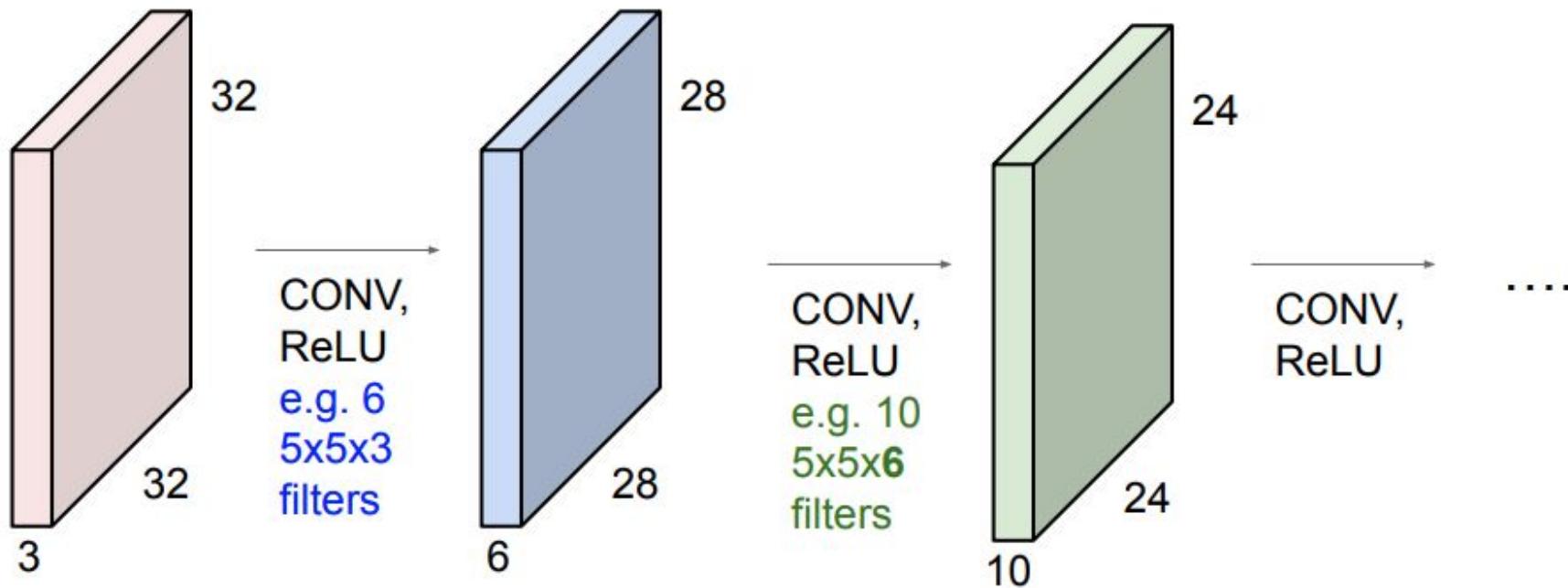
Sursă foto: [https://en.wikipedia.org/wiki/Kernel_\(image_processing\)#/media/File:2D_Convolution_Animation.gif](https://en.wikipedia.org/wiki/Kernel_(image_processing)#/media/File:2D_Convolution_Animation.gif)

Convoluție

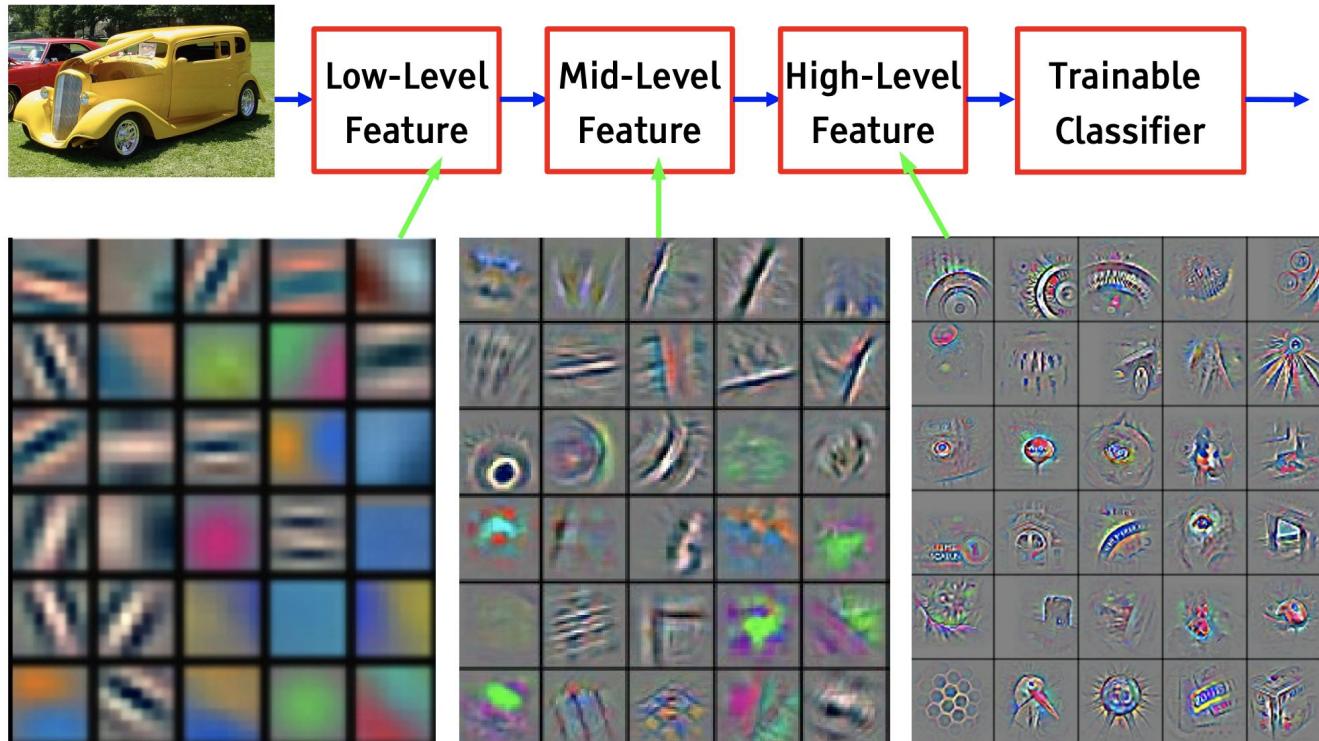
- Un filtru produce un singur activation map
- Mai multe filtre produc mai multe activation maps - un volum similar cu imaginea de input
- Fiecare filtru învață un aspect (feature) diferit din input



Convoluție



Reprezentare ierarhică



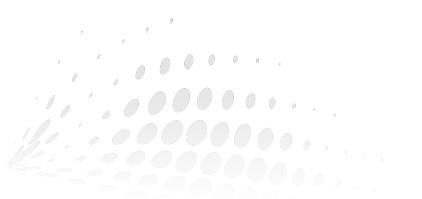
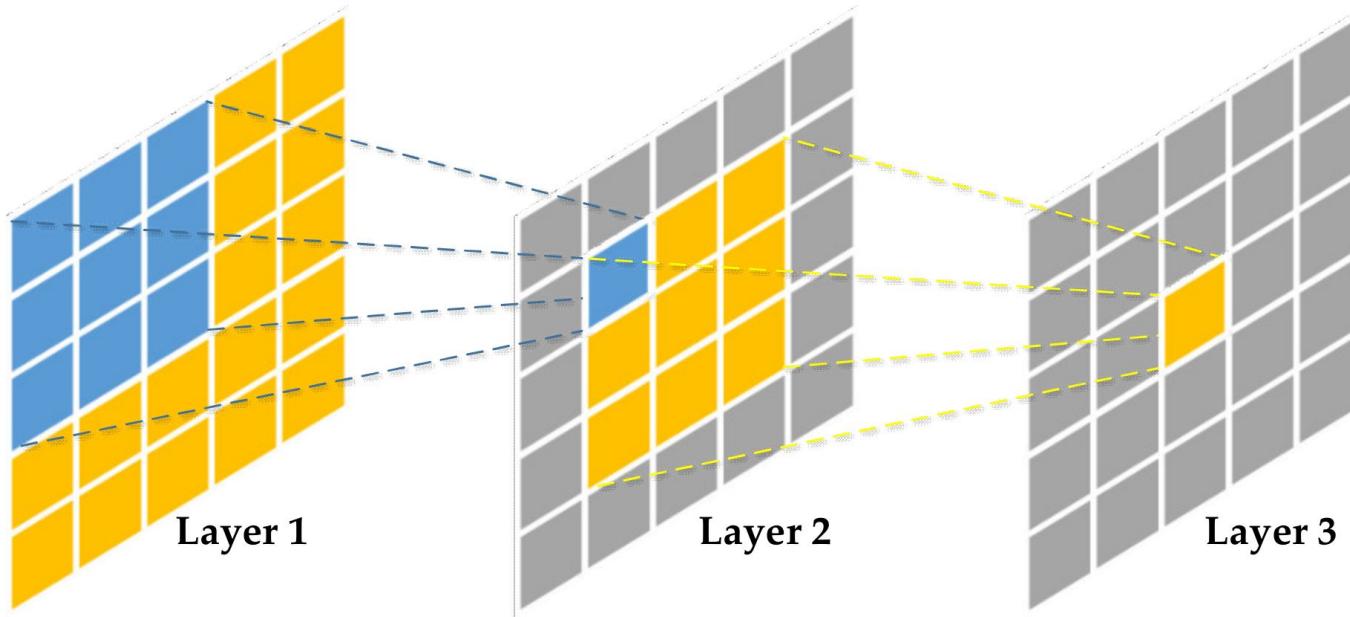
Feature visualization of convolutional net trained on ImageNet from [Zeiler & Fergus 2013]

Convoluție - Conectivitate Locală

- în cazul imaginilor de dimensiuni mari, este nefezabilă conectarea neuronilor cu toți neuronii din volumul anterior
- neuronii sunt conectați doar la o regiune locală din volumul anterior (receptive field, echivalent cu dimensiunea filtrului)
- dimensiunea în adâncime a filtrului de conoluție este întotdeauna egală cu adâncimea volumului de input (conoluția este locală pe lățime și adâncime, dar completă pe adâncime)



Convoluție - Conectivitate Locală



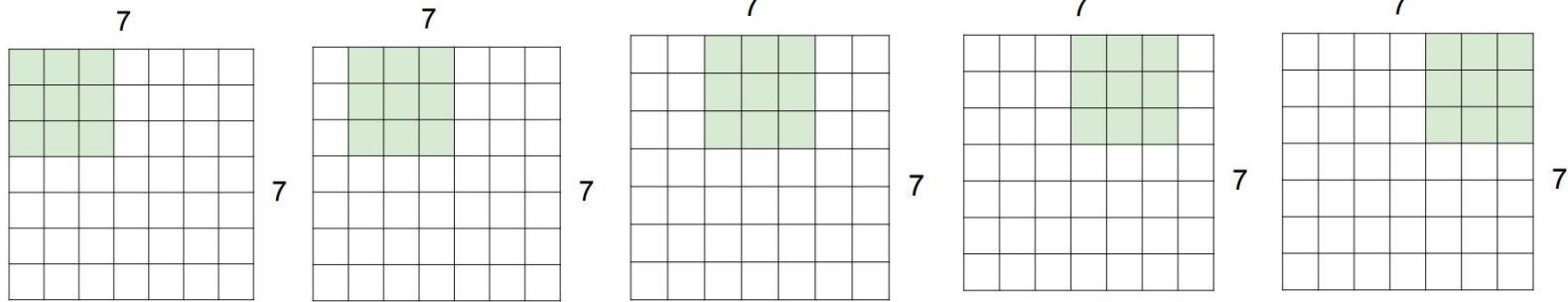
Convoluție - Hiperparametri

- **depth** - corespunde numărului de filtre pe care vrem să le folosim, fiecare filtru activându-se la un feature diferit din input
- **stride** - pasul cu care glisăm filtrul; pentru stride 1, glisăm filtrul cu câte un pixel; pentru stride k, glisăm filtru cu câte k pixeli, rezultând un volum mai mic din punct de vedere spațial decât volumul de input
- **padding** - păstrează dimensiunea spațială a volumului de input. Nu vrem sa micsoram volumul prea mult si prea repede.



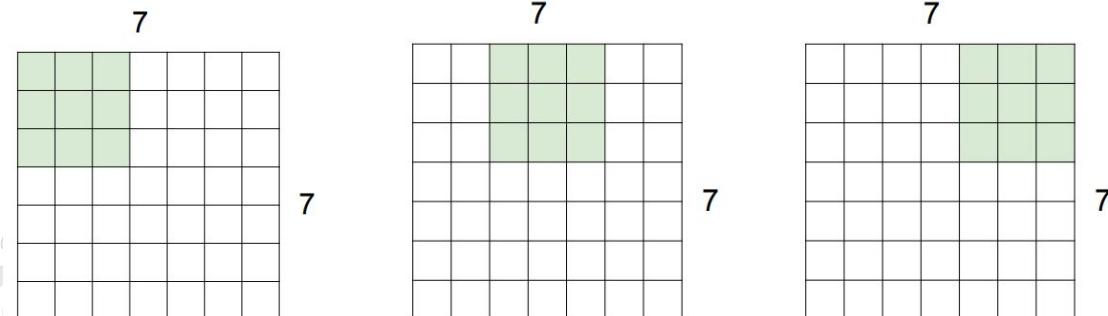
Stride

Stride 1



=> 5x5 output

Stride 2

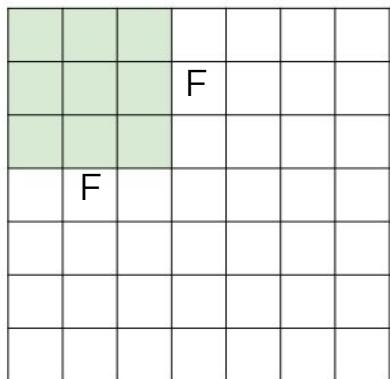


=> 3x3 output

Stride

Ce se intampla daca vrem sa folosim stride = 3?

7 =N

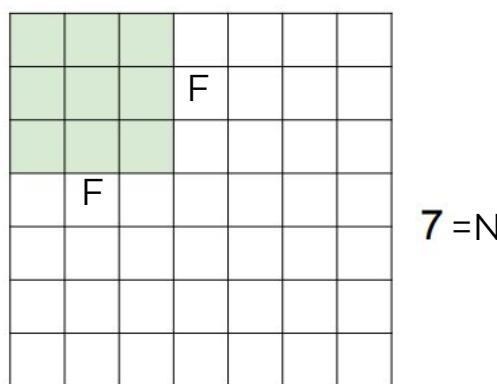


7 =N



Stride

Ce se intampla daca vrem sa folosim stride = 3?



Dimensiune output-ului = **(N-F) / stride + 1**

Ex.: N = 7, F = 3:

$$\text{Stride 1} \Rightarrow (7 - 3)/1 + 1 = 5 \quad \text{OK}$$

$$\text{Stride 2} \Rightarrow (7 - 3)/2 + 1 = 3 \quad \text{OK}$$

$$\text{Stride 3} \Rightarrow (7-3)/3 + 1 = 2.33 \quad \text{NOT OK}$$

Padding

0	0	0	0	0	0			
0								
0								
0								
0								



Padding

0	0	0	0	0	0			
0								
0								
0								
0								

Ex.: Pentru un input 7×7 si un filtru 3×3 cu stride = 3 si padding = 1, care va fi dimensiunea output-ului?



Padding

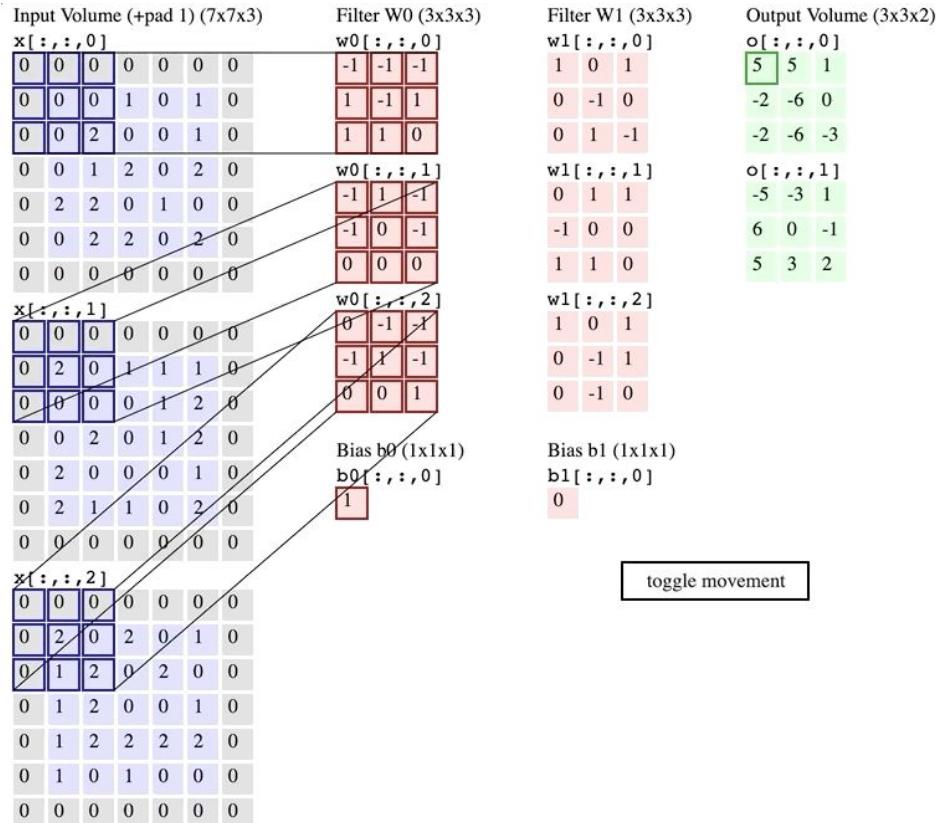
0	0	0	0	0	0			
0								
0								
0								
0								

Ex.: Pentru un input 7x7 si un filtru 3x3 cu stride = 3 si padding = 1, care va fi dimensiunea output-ului?

Formula: **(N + 2P - F) / stride + 1**



Convoluție - Demo



Strat Convoluțional - Numarul parametrilor

?



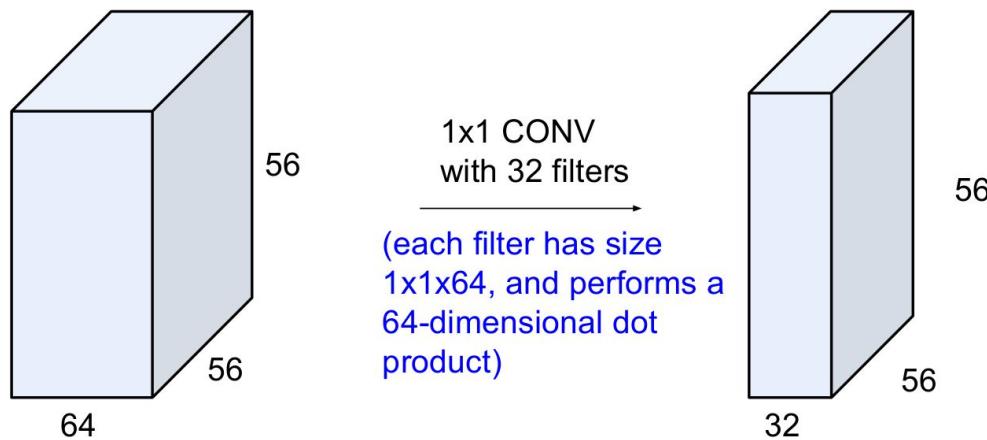
Strat Convoluțional - Numarul parametrilor

$F \times F \times D \times K$ weights + K biases



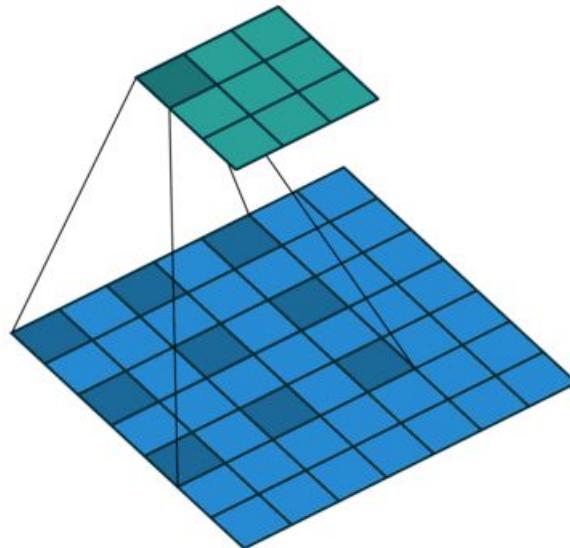
Convoluția 1x1

- afectează doar adâncimea volumului
- rol în reducere a dimensionalității



Convoluția Dilatată

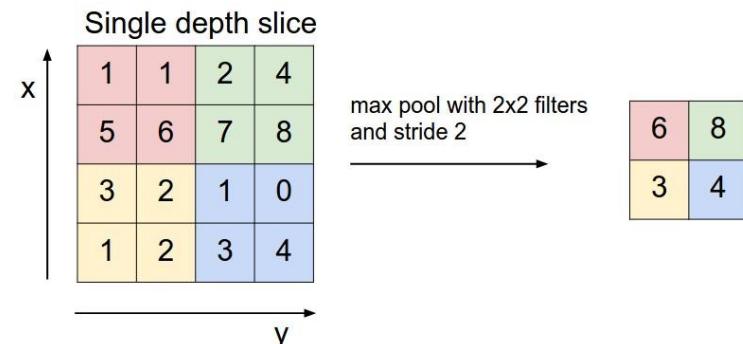
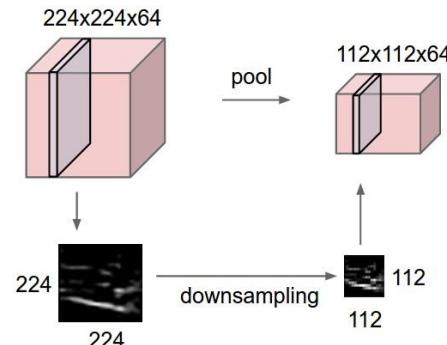
- filtrele au spații între elemente
- folosite pentru a mări receptive field-ul



https://miro.medium.com/max/790/0*3cTXlemm0k3Sbask.gif

Pooling

- reduce dimensiunea spațială (înălțime și lățime) a stratului precedent
- operează independent pe fiecare nivel din adâncime
- tipuri de pooling: MAX, AVG, L2-norm
- cea mai comună formă de aplicare este cu filtre de dimensiune 2×2 și stride 2
- alternativă: convoluție cu stride 2



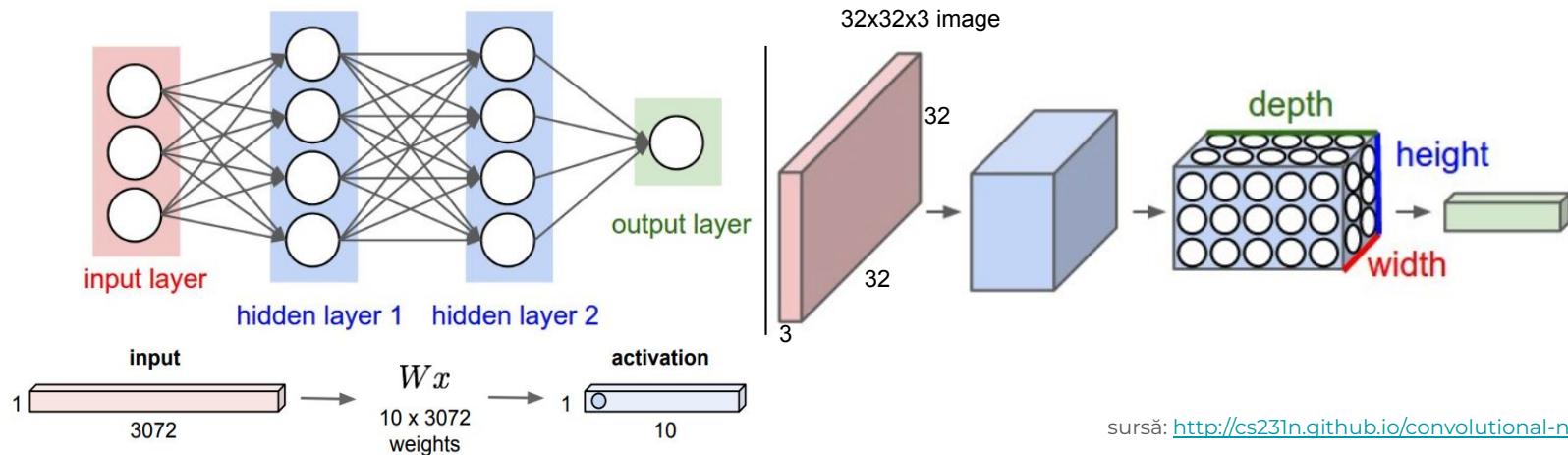
sursă: <http://cs231n.github.io/convolutional-networks/>

Convoluție - Partajarea parametrilor

- numărul de parametri este redus drastic dacă se face următoarea asumție rezonabilă: dacă un feature este suficient de bun pentru a fi calculat la poziția (x_1, y_1) , atunci ar trebui să fie util și la poziția (x_2, y_2) . Astfel constrângem toți neuronii de la un anumit nivel din adâncime dintr-un strat de conoluție (e.g. un volum de dimensiune $[55 \times 55 \times 96]$ are 96 de niveluri în adâncime) să folosească aceiași parametri.
- asumția anterioară este validă datorită structurii de invariантă la translație a imaginilor



Asemănări cu rețelele tradiționale



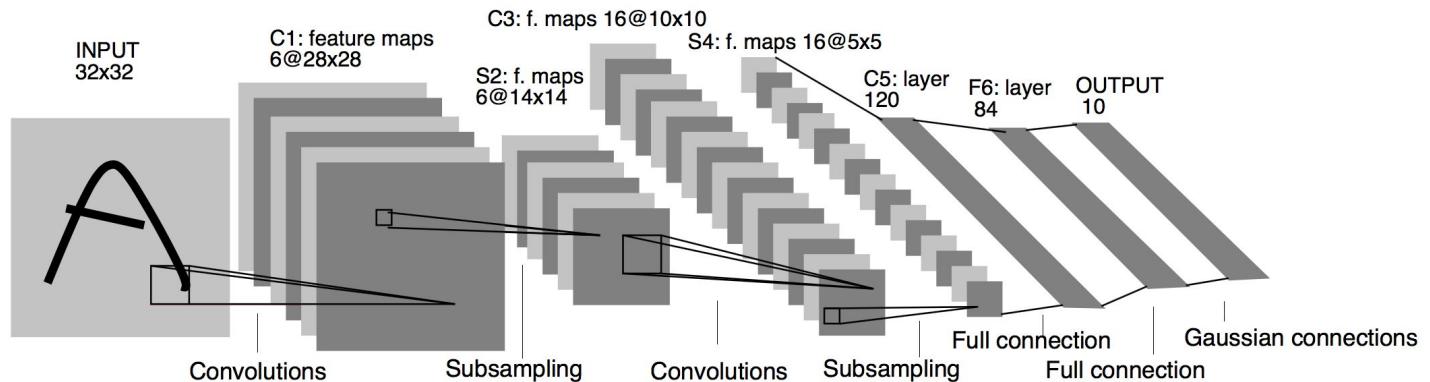
- sunt alcătuite din neuroni care au parametri și bias învățabili
- fiecare neuron primește un input, efectuează un produs scalar și optional aplică o funcție neliniară
- au o funcție de cost pe ultimul strat și conceptele de bază ale rețelelor tradiționale se aplică în continuare

Diferențe față de rețelele tradiționale

- fac presupunerea explicită că input-urile sunt imagini, ceea ce ne permite să introducem anumite proprietăți în arhitectura rețelei, astfel că se reduce foarte mult numărul de parametri
- neuronii straturilor sunt aranjați în 3 dimensiuni (lățime, înăltime, adâncime) și sunt conectați doar cu o mică **regiune** din stratul anterior



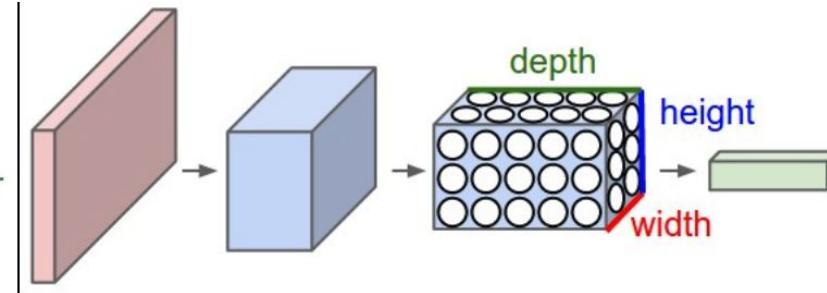
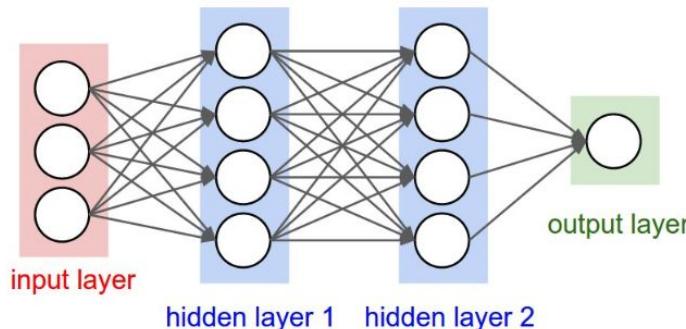
Rețele Neurale Convoluționale - Structură



Y. Lecun, L. Bottou, Y. Bengio and P. Haffner, "Gradient-based learning applied to document recognition," in Proceedings of the IEEE, vol. 86, no. 11, pp. 2278-2324, Nov. 1998, doi:10.1109/5.726791.

- INPUT - structură 3D, conține valorile pixelilor imaginii
- CONV - calculează output-ul neuronilor conectați la regiuni din volumul anterior
- RELU - funcție de activare a fiecărui element
- POOL - reduce dimensiunea input-ului de-a lungul lățimii și înăltimii
- FC - calculează scorurile claselor; fiecare neuron este conectat la toți neuronii din volumul anterior

Avantaje CNN



sursă: <http://cs231n.github.io/convolutional-networks/>

Fully-Connected:

- nu scalează pentru imagini mari
- ignoră informația spațială (poziția pixelilor și corelația cu vecinii)
- nu sunt invariabile la translație

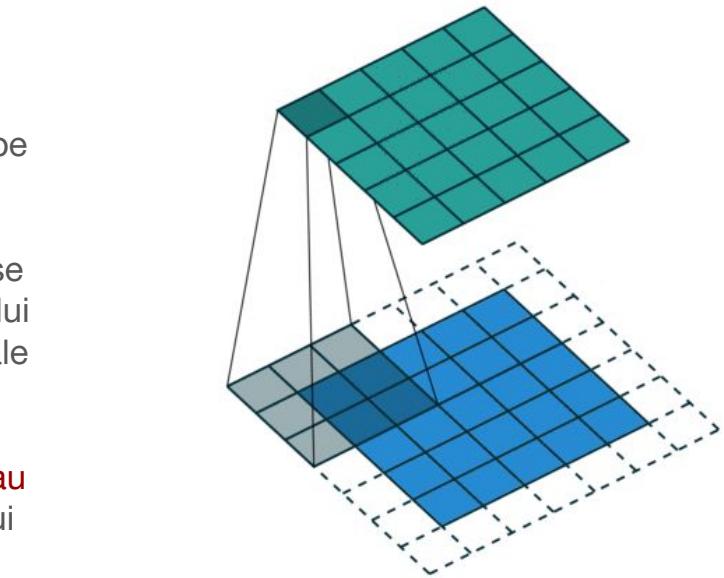
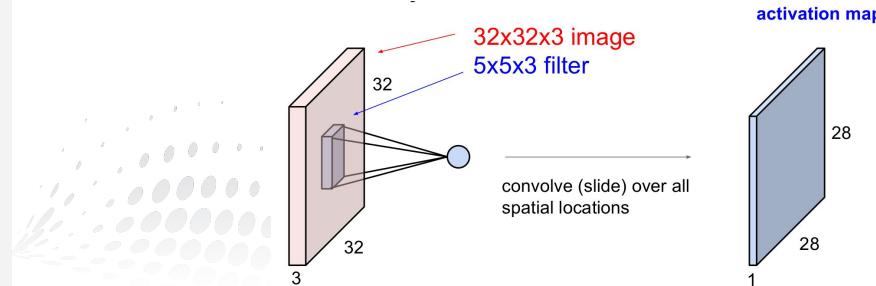
CNN:

- poziția pixelilor și vecinii conțin informație semantică
- elementele de interes pot apărea oriunde în imagine



Recapitulare convolutii (1)

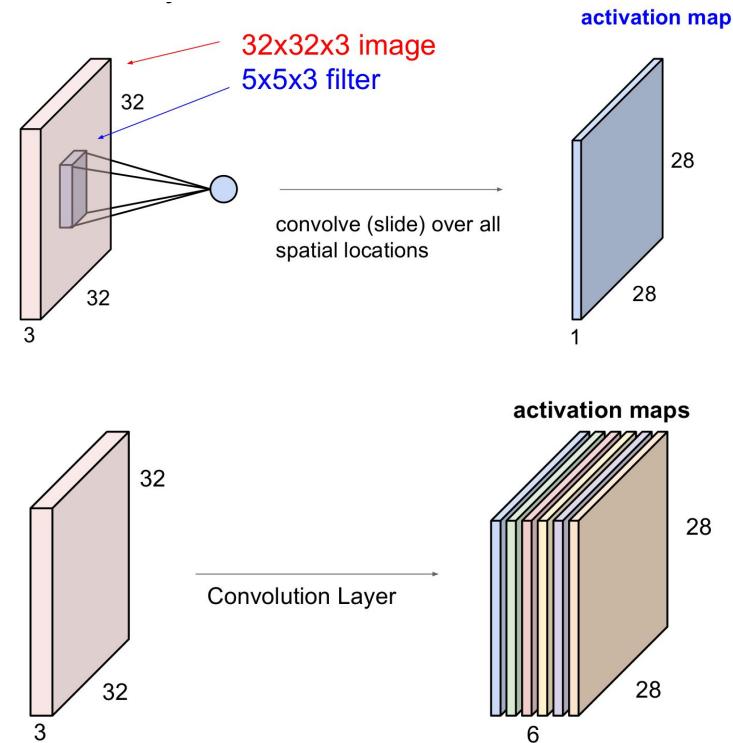
- **Convolutie:** Filtru / kernel cu parametri invatabili.
- **Dimensiuni standard:** 1x1, 3x3, 5x5, 7x7. Filtrele se extind pe toata adancimea volumului de intrare. (e.g. 5x5x3)
- **Rezultatul operatiei de convolutie:** In timpul antrenarii, se gliseaza fiecare filtru de-a lungul latimii si lungimii volumului anterior si se calculeaza produsul scalar intre elementele sale si regiunea corespunzatoare din input, la fiecare pozitie.
- Rezultatul convolutiei este o harta 2D (**activation map sau feature map**), reprezentand raspunsul aplicarii filtrului respectiv la fiecare pozitie din input.



Sursa: http://deeplearning.net/software/theano/_images/same_padding_no_strides.gif

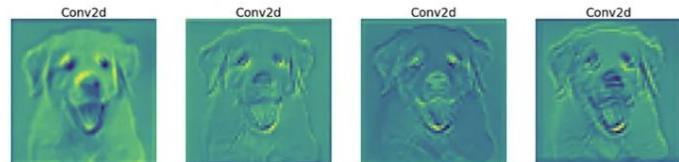
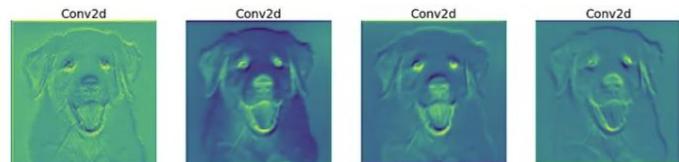
Recapitulare convolutii (2)

- Un singur filtru produce un singur activation map.
 - Daca folosim un singur filtru, atunci outputul isi pierde din volum.
- Mai multe filtre genereaza mai multe activation maps.
 - Avem, asadar, control asupra volumului outputului.



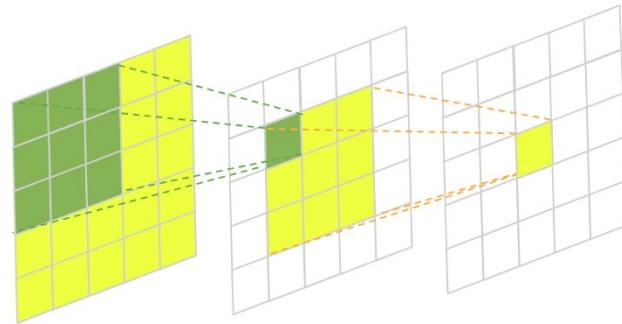
Recapitulare convolutii (3)

- Avantajele folosirii mai multor filtre in cadrul unei convolutii
 - Fiecare filtru va invata trasaturi diferite ale inputului.
- De ce ajuta invataarea mai multor trasaturi?
 - Modelul capata abilitatea de a invata trasaturile care ajuta la rezolvarea problemei.
 - Filtrele pastreaza proprietatea de **conectivitate locala**, ceea ce inseamna ca si trasaturile generate vor avea aceasta proprietate.

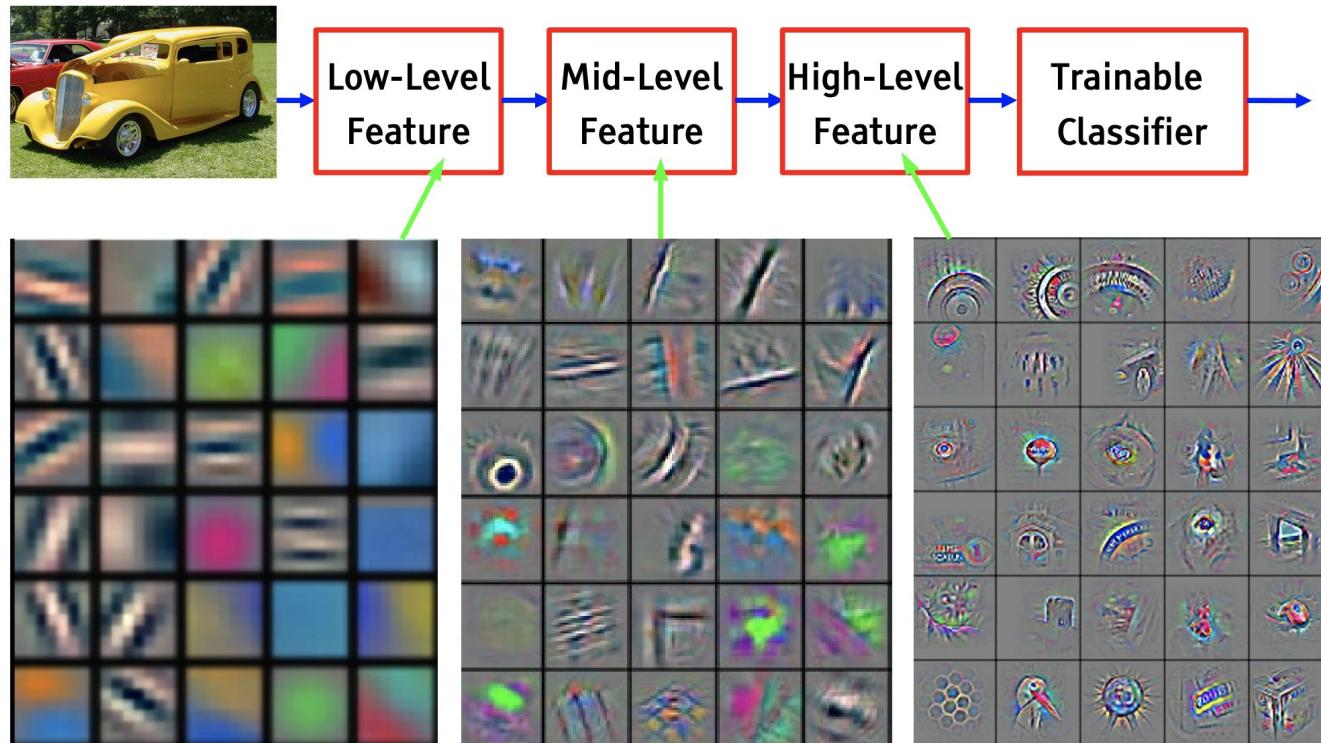


Recapitulare convolutii (4)

- **Receptive field**
 - Pentru un neuron dintr-un strat de convolutie, notiunea de receptive field se refera la regiunea din inputul original la care neuronul este sensibil.
 - Cu alte cuvinte: ce regiune din input are influenta asupra rezultatului dat de un anumit neuron?
- **Ce se intampla pe masura ce avansam in retea?**
 - Receptive field-ul neuronilor creste.
- **Consecinta:**
 - Filtrele din retea invata trasaturi din ce in ce mai complexe cu cat avansam in adancimea ei.



Recapitulare convolutii (5) - Reprezentare ierarhica

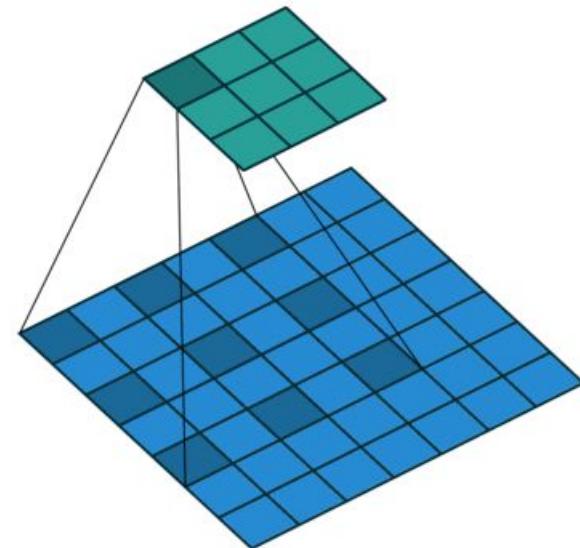


Feature visualization of convolutional net trained on ImageNet from [Zeiler & Fergus 2013]

Sursă slide: <https://www.di.ens.fr/willow/events/cvml2013/materials/slides/tuesday/lecun-20130723-vrml-01.pdf>

Recapitulare convolutii (6) - convolutia dilatata

- **Convolutie dilatata**
 - Convolutie in care filtrele au weight-urile spatiate.
- **Rata de dilatare**
 - Determina distanta dintre weight-urile filtrului.
 - Exemplul din dreapta: $d = 2$
 - Convolutia normala are rata de dilatare $d = 1$.
- **Cand este folosita?**
 - Atunci cand vrem ca receptive field-ul sa creasca mai repede.
 - Rata de dilatare nu trebuie sa fie nici prea mare, pentru ca se pierde proprietatea de conectivitate locala.



Recapitulare convolutii (7) - pooling

- **Motivatie**

- Reducerea dimensiunii hartilor de activare pe lungime si latime, in timp ce pastram numarul de canale (adancimea ramane neschimbata).

- **Cum opereaza?**

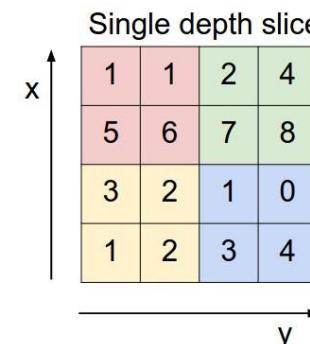
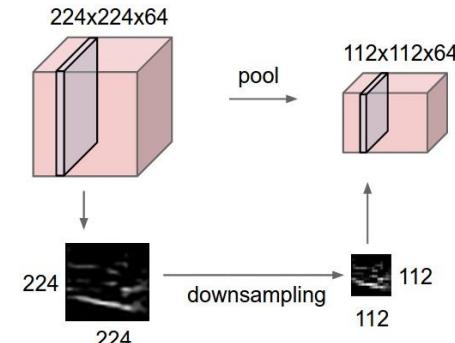
- Independent pe fiecare nivel de adancime.
- Cu alte cuvinte: fiecarei harti de activare anterioare i se reduce in mod independent dimensiunea spatiala.

- **Tipuri de operatii folosite**

- Max, Average, L2 norm

- **Dimensiune folosita in mod normal**

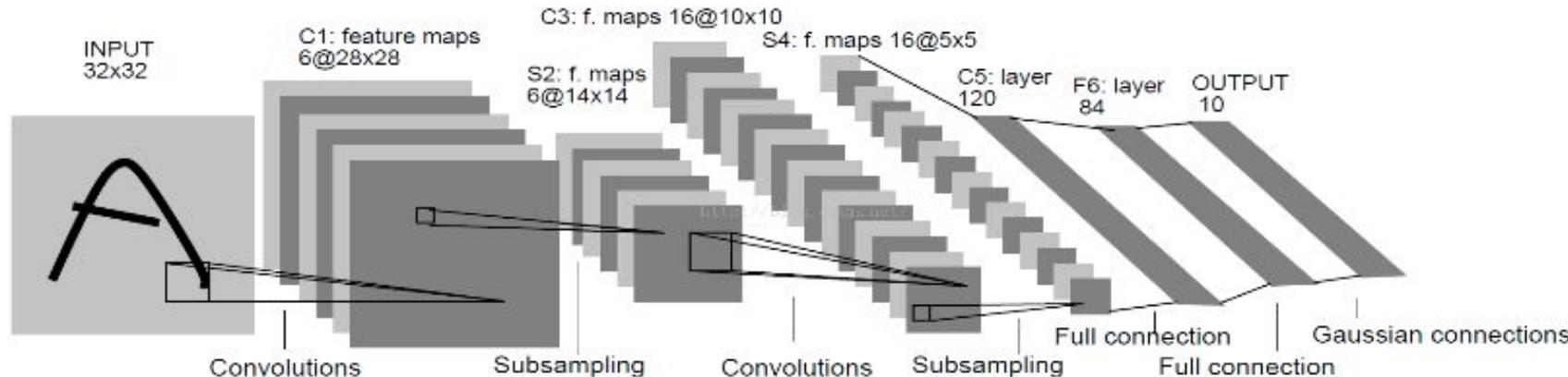
- In mod normal straturile de pooling au dimensiuni de 2×2 si stride 2.



sursă: <http://cs231n.github.io/convolutional-networks/>

LeNet-5 [LeCun et al., 1998]

- 2 straturi convolutionale 5×5 cu stride 1
- 2 operatii de average-pooling 2×2 cu stride 2
- 7 straturi antrenabile
- > 98.5 % accuracy on MNIST



ImageNet - challenge

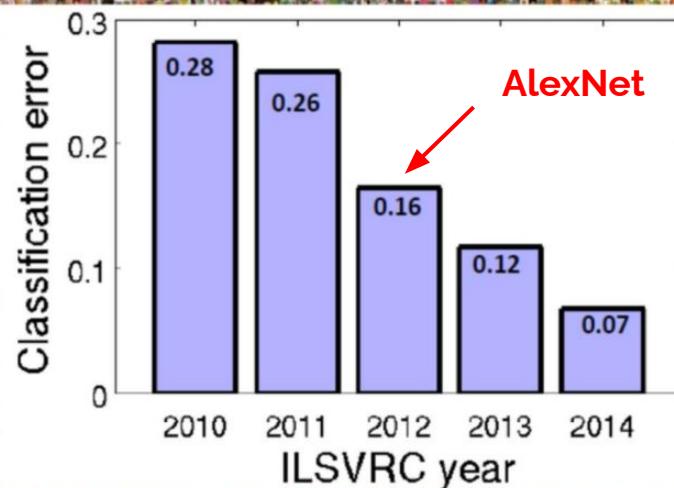
- Peste 14M de imagini de diferite dimensiuni.
- ~22K categorii corespunzatoare unor synset-uri din WordNet
- Ierarhie semantica
- Colectate de pe internet si etichetate manual (Amazon Mechanical Turk)



IMAGENET Large Scale Visual Recognition Challenge

Steel drum

The Image Classification Challenge:
1,000 object classes
1,431,167 images



Russakovsky et al. arXiv, 2014

ImageNet ILSVRC

ImageNet Large Scale Visual Recognition Challenge

- Competitie de clasificare si localizare
 - 1000 de clase
 - ~1M imagini de antrenare
 - 50K imagini de validare (publice)
 - 150K imagini de evaluare
- Evaluare acuratete top-5
 - Categoria corecta este in primele 5 categorii prezise?
- Pana in 2012, ~25% eroare decenta
- Deep revolution (2012): breakthrough ~16 % AlexNet
- Eroare umana: 5.1% (3.6% ansamblu)

ImageNet ILSVRC

Clase diferite

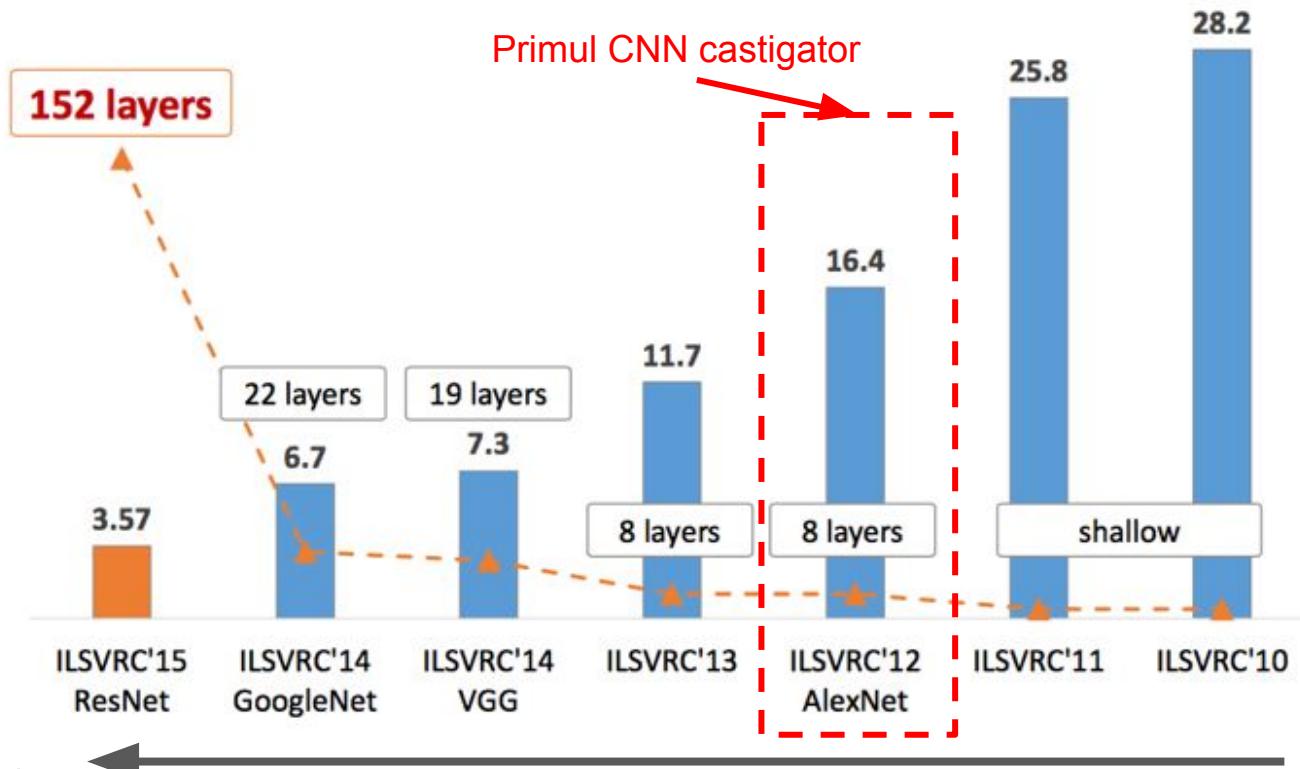


(a) Siberian husky



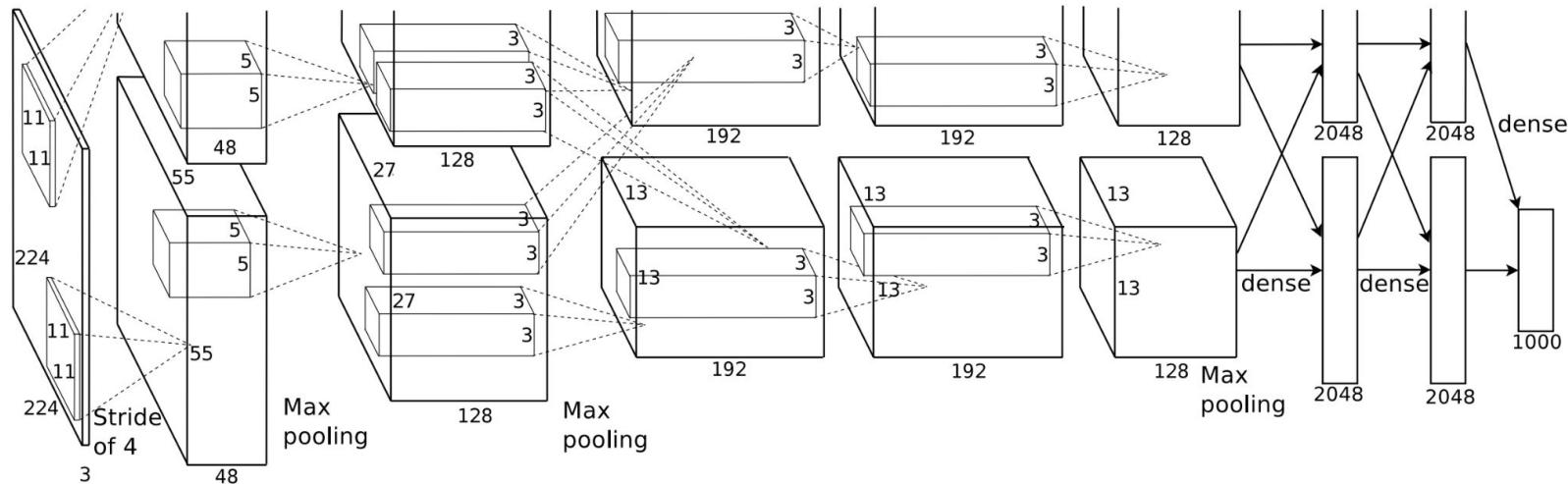
(b) Eskimo dog

Castigatorii ILSVRC



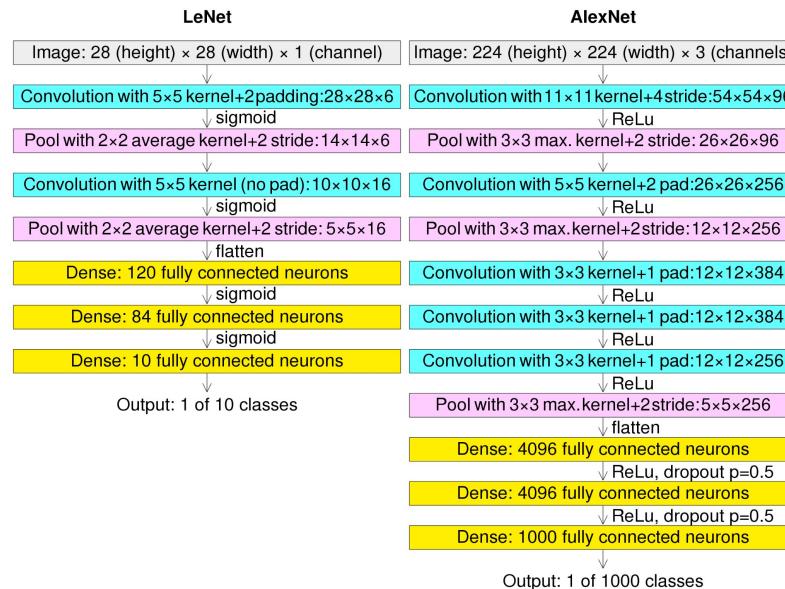
AlexNet [Krizhevsky et al. 2012]

- **Schimbari:**
 - ReLU
 - Antrenare pe 2 GPU-uri (limitate la 3GB de memorie fiecare)
 - Local Response Normalization
 - Augmentari
 - Dropout

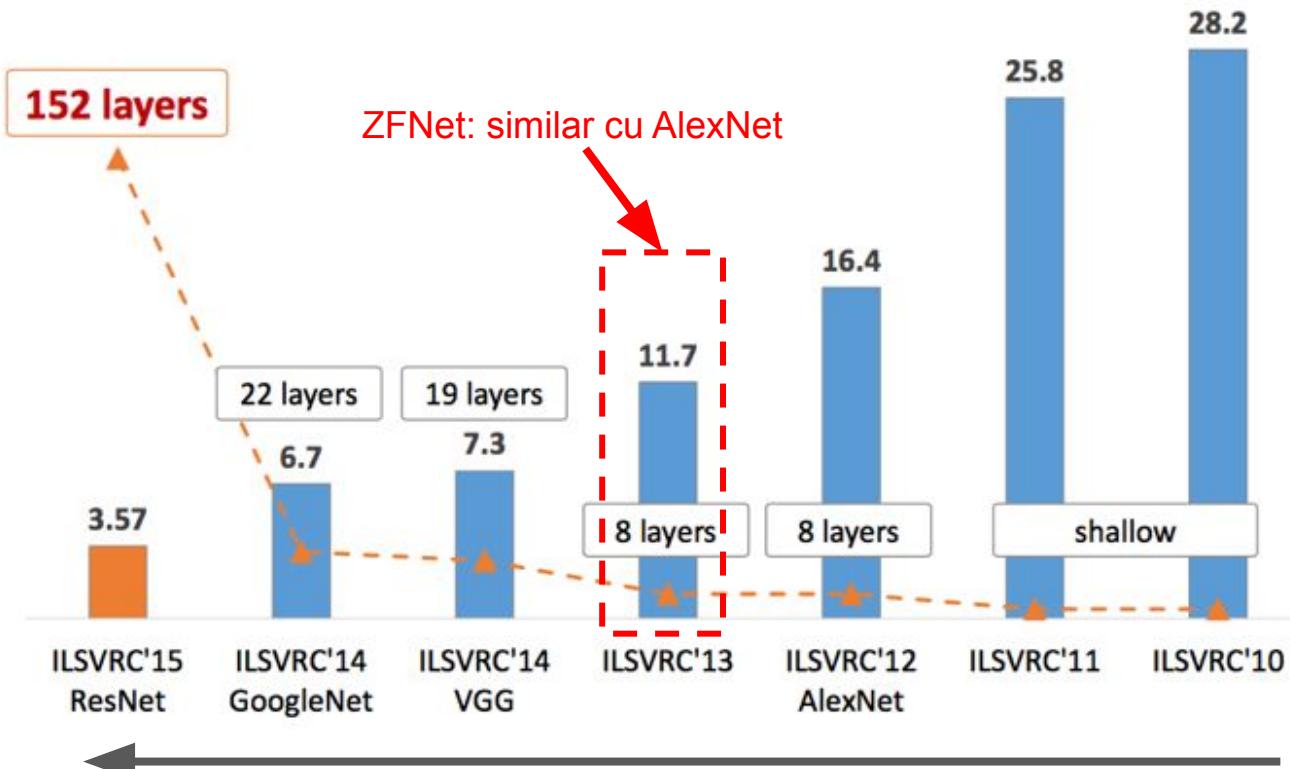


AlexNet [Krizhevsky et al. 2012]

- Schimbari:
 - ReLU
 - Antrenare pe 2 GPU-uri (limitate la 3GB de memorie fiecare)
 - Local Response Normalization (“brightness normalization”)
 - Augmentari
 - Dropout

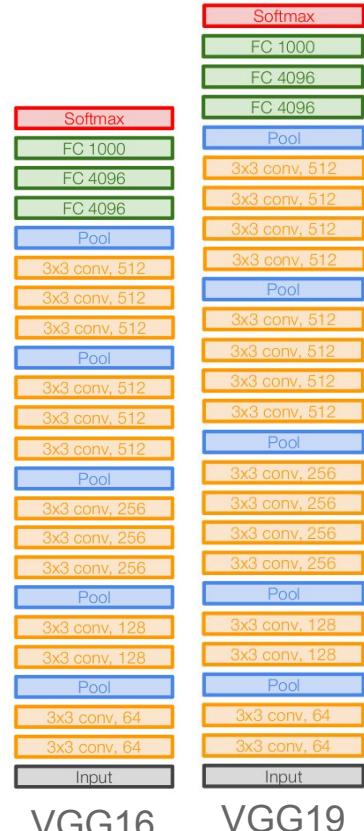


Castigatorii ILSVRC



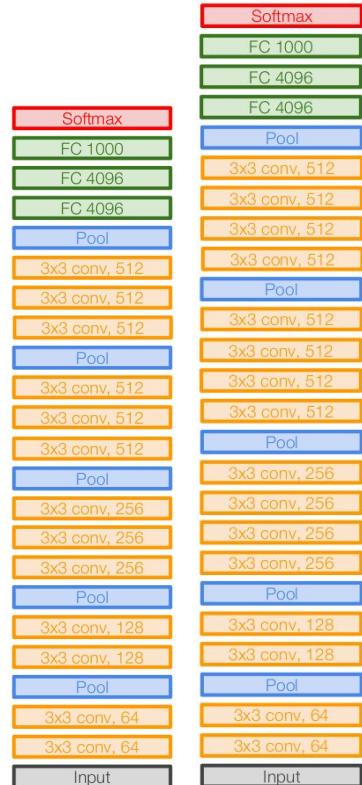
VGGNet [Simonyan and Zisserman, 2014]

- Runner-up in 2014 pentru task-ul de clasificare si castigator pe restul task-urilor
- 7.3 % eroare top-5 pe ILSVRC 2014
- **Aspecte cheie:**
 - Retea mai adanca (deep)
 - Arhitectura simplificata
 - Doar conv 3x3 cu stride 1, pad 1
 - MaxPooling 2x2 cu stride 2
 - VGGNet-16: 13 CONV 3x3 + 3 FC
 - VGGNet-19: 16 CONV 3x3 + 3 FC
- VGG19 e doar putin mai bun



VGGNet [Simonyan and Zisserman, 2014]

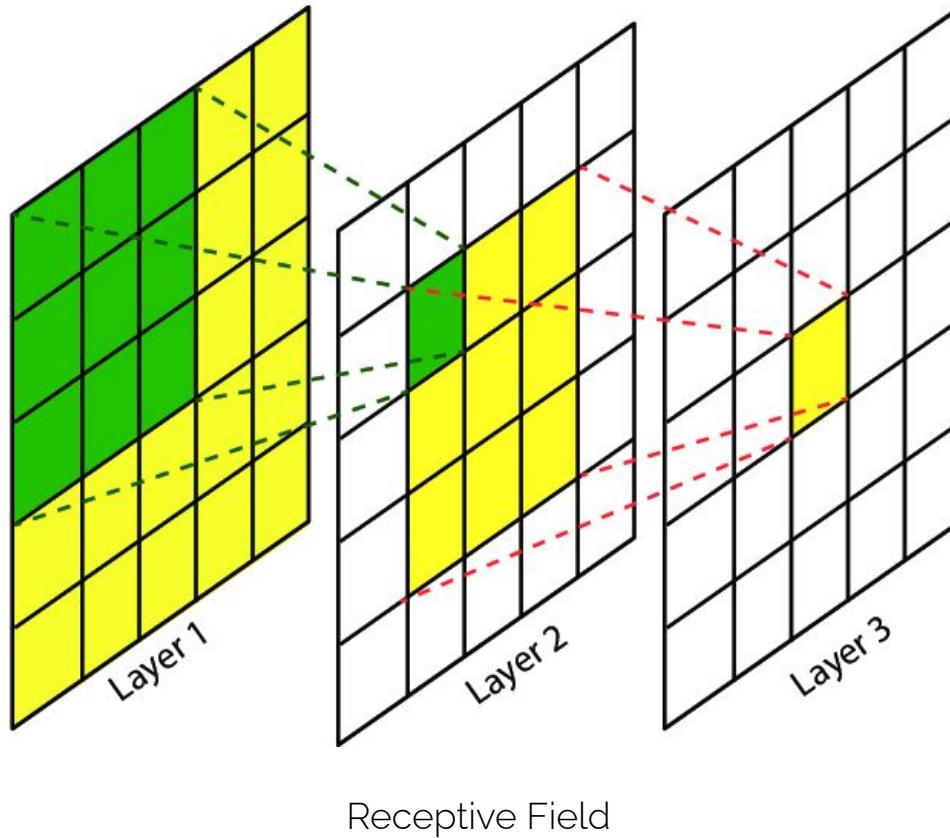
- De ce conv mai mici 3x3?
 - Convolutiile mari sunt ineficiente. 3x3 e lightweight.
 - Mai deep decat 7x7.
 - Au mai multe non-linearitati decat un singur 7x7.
 - 3 conv 3x3 stack-uite au un *receptive field* echivalent cu cel al unui singur conv 7x7.



VGG16

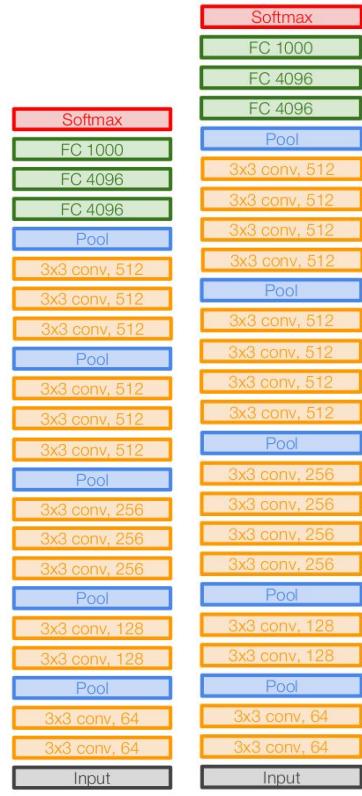
VGG19

VGGNet [Simonyan and Zisserman, 2014]

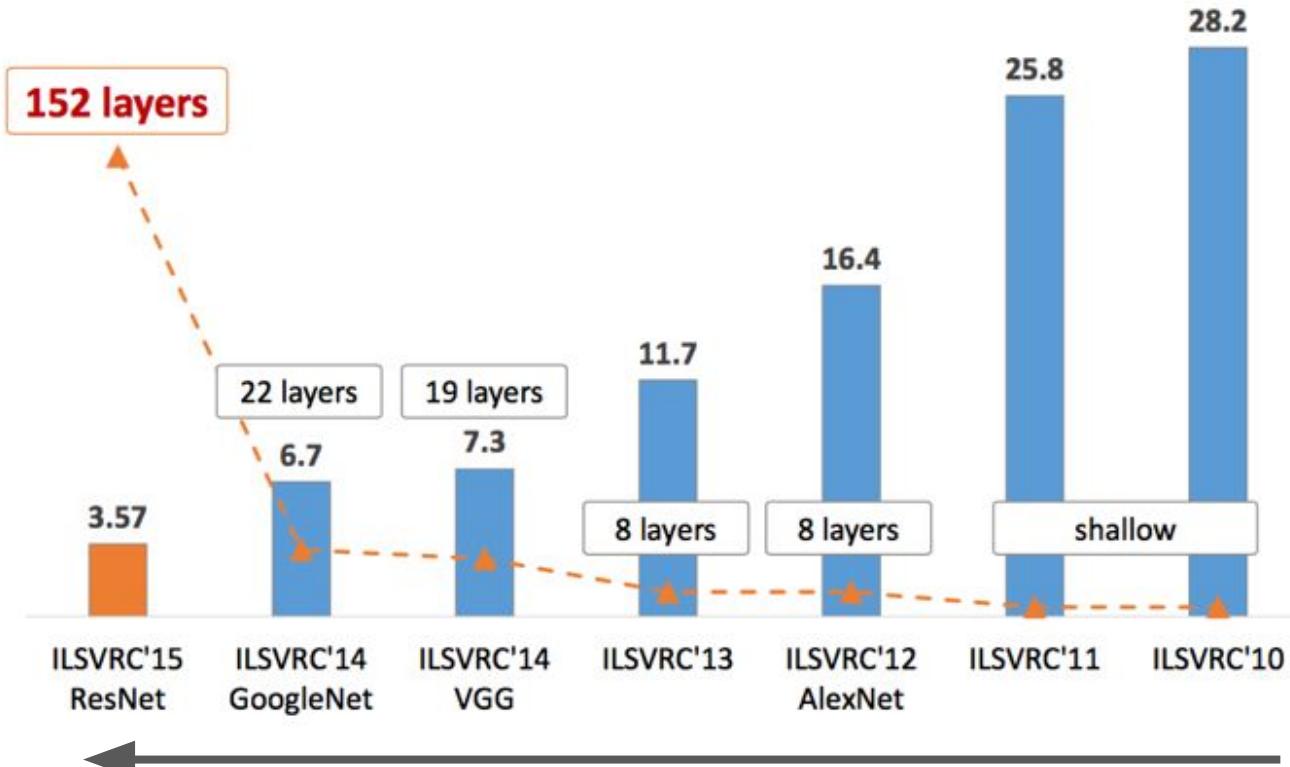


VGGNet [Simonyan and Zisserman, 2014]

- De ce conv mai mici 3x3?
 - Convolutiile mari sunt ineficiente. 3x3 e lightweight
 - Mai deep decat 7x7
 - Au mai multe non-linearitati decat un singur 7x7
 - 3 conv 3x3 stack-uite au un *receptive field* echivalent cu cel al unui singur conv 7x7.
 - Pastrand constant pe nivel depth-ul D:
 - $3 \times (3 \times 3 \times D \times D)$ parametri pentru 3 conv 3x3
 - $1 \times (7 \times 7 \times D \times D)$ parametri pentru 1 conv 7x7
 - $(7 \times 7) / (3 \times 3 \times 3) \sim 1.81$ mai multi parametri pt 7x7

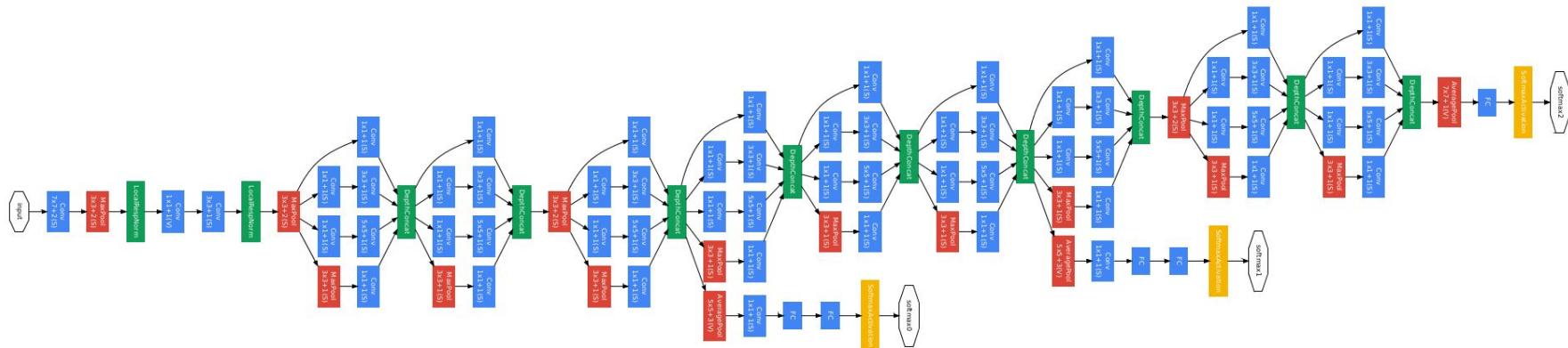


Castigatorii ILSVRC



GoogLeNet (Szegedy et al., 2014)

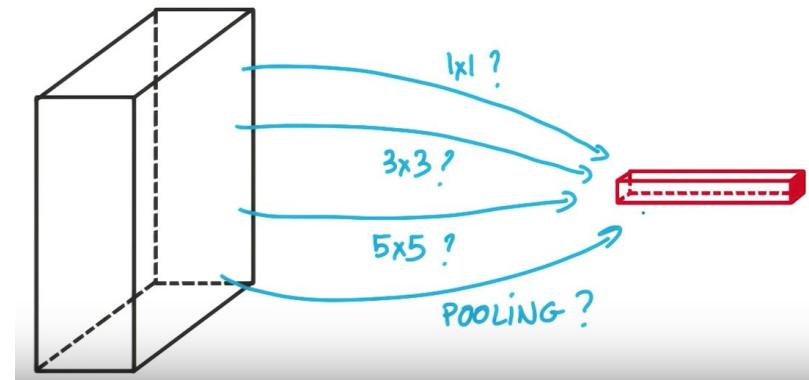
- 6.66% eroare de clasificare top 5 pe ImageNet
 - Modulul *Inception* (practic o mini-retea în cadrul unei rețele mai mari)



http://joelouismarino.github.io/images/blog_images/blog_qooglenet_keras/qooglenet_diagram.png

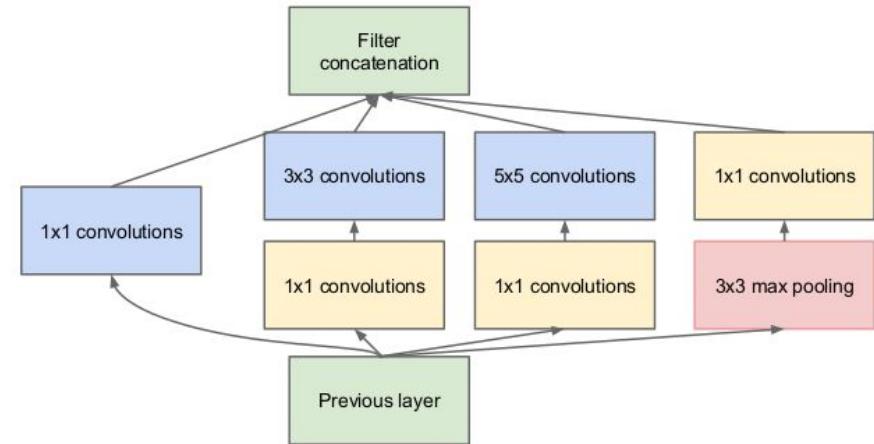
Modulul *Inception*

- **Problema** pe care incearca sa o rezolve:
 - Necesitatea alegerii tipului de convolutie la nivelul fiecărui layer (3×3 , 5×5 ...)
- **Solutie:**
 - Le folosim pe toate si lasam reteaua sa decida.
 - Adaugam convolutiile in paralel si concatenam rezultatele inainte de a trece la nivelul urmator.



Modulul *Inception* - Arhitectură

- Varietate de convolutii: 1x1, 3x3, 5x5
- **Convolutii 1x1** înaintea convolutiilor ‘mari’ (3x3, 5x5), pentru **reducerea dimensionalitatii**
- Maxpooling 3x3 din *considerente istorice* (toate arhitecturile bune aveau pooling)



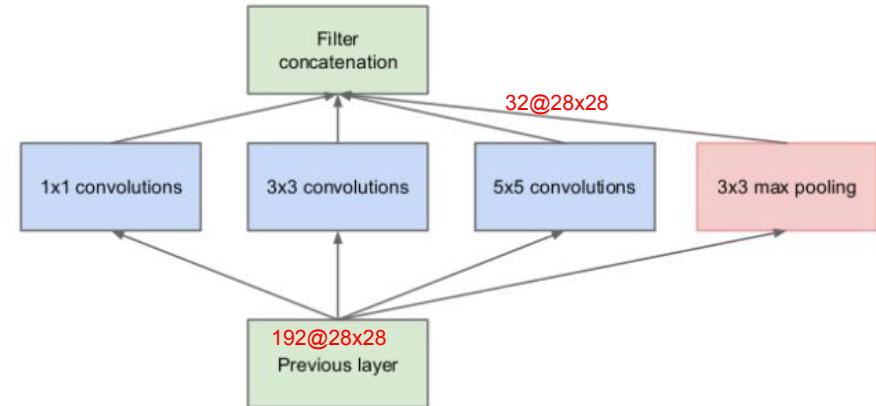
Modulul *Inception* - Arhitectură naivă

- Exemplu: care este numărul de operații pe ramura 3 (convoluție 5x5)?

- **Formula:**

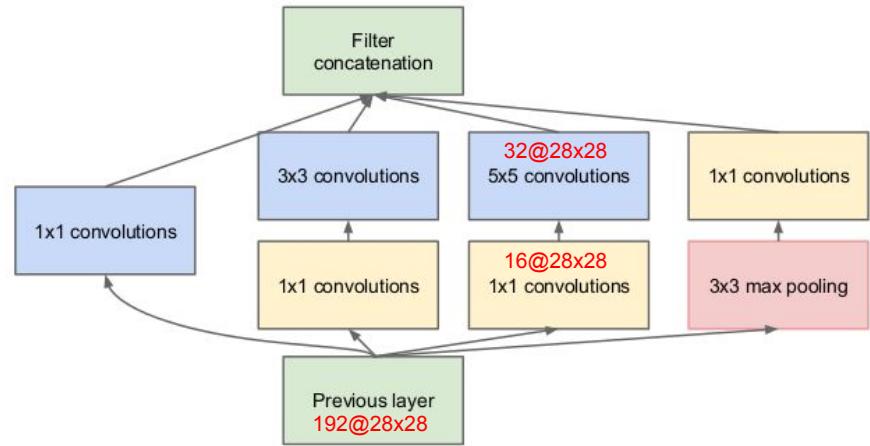
- $C_0 * C_1 * H * W * F * F$
- C_0 = # canale înainte de aplicarea convoluției
- C_1 = # canale după aplicarea convoluției
- H, W = lungimea și latimea activation map-ului rezultat
- F = dimensiunea filtrului

- $5 * 5 * 28 * 28 * 192 * 32 = 120M$



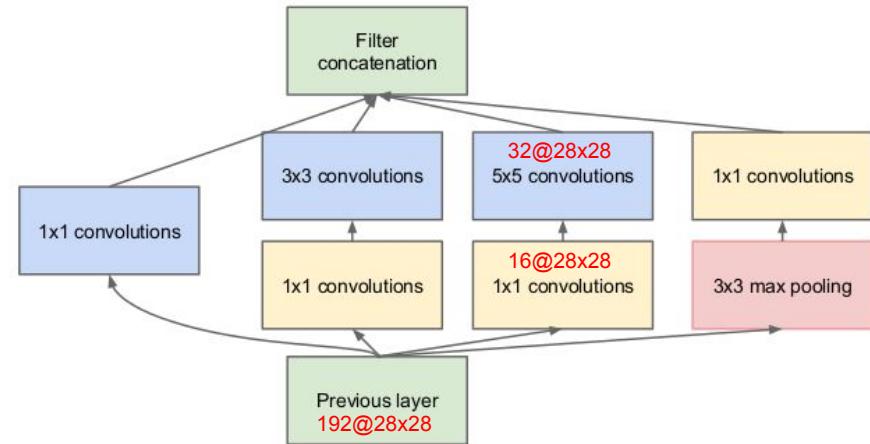
Modulul *Inception*

- Exemplu: care este numărul de operații pe ramura 3 (conoluție 5x5)?

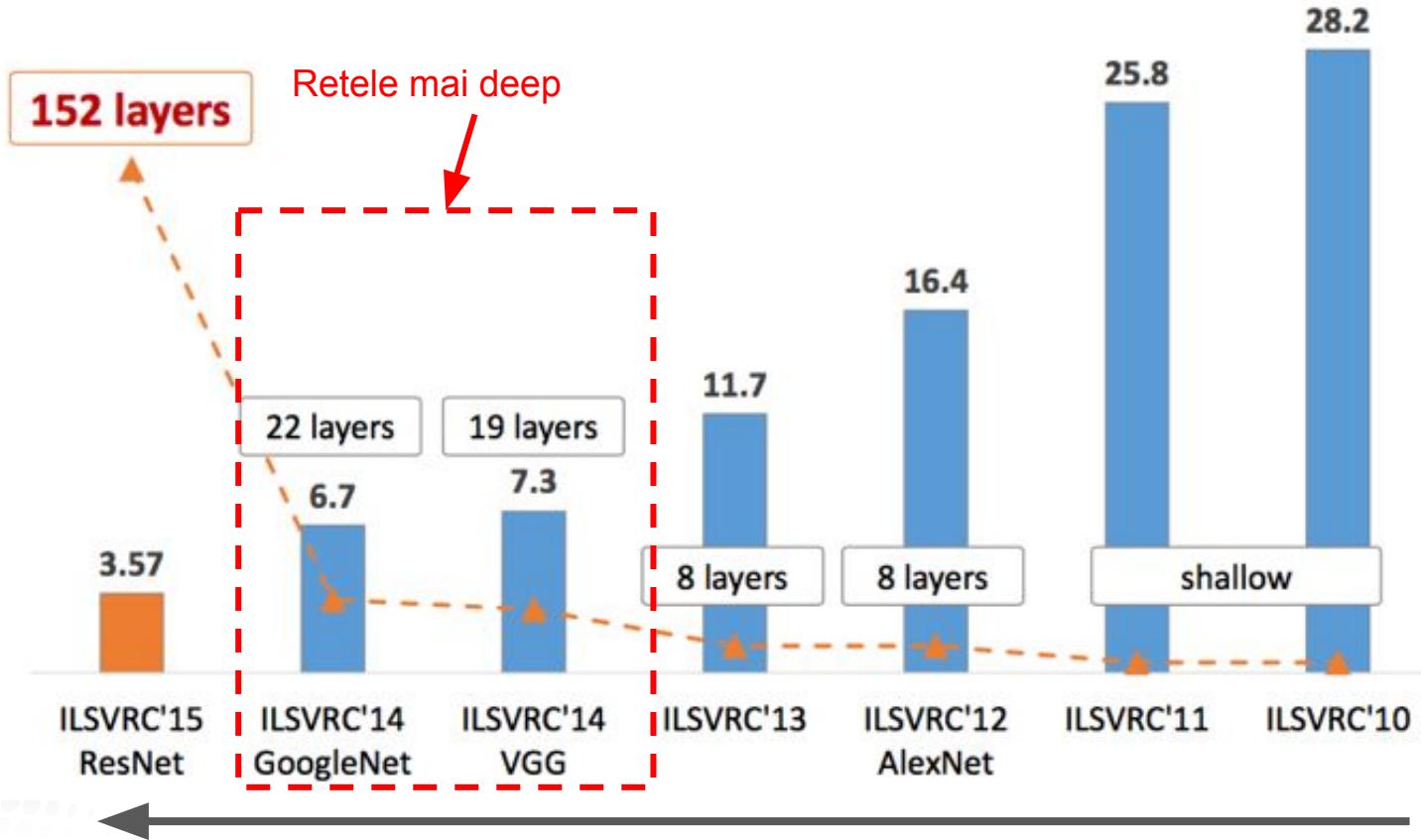


Modulul Inception

- Exemplu: care este numărul de operații pe ramura 3 (convoluție 5x5)?
- $1*1*28*28*192*16 + 5*5*28*28*16*32 = 12M$



Castigatorii ILSVRC (2014)



Standardizare/Normalizare

- **Motivatie:**
 - Stabilitate numérica
 - Feature-urile au o importanță similară
 - Antrenare mai rapidă
 - E.g. ImageNet mean = [0.485, 0.456, 0.406] and std = [0.229, 0.224, 0.225]

Batch Normalisation (1)

- BatchNorm-ul este un strat in retea care reprezinta o metoda eficiente de regularizare
 - Activari curate
 - Stabilitatea antrenarii
 - Rate de invatare mai mari
 - Antrenare mai rapida
- Re-normalizeaza outputul unui strat in pregatire pentru urmatorul.
- Decoupleaza straturile prin normalizare.
- Biasurile din straturi nu mai sunt folositoare (translatia din BN preia acest rol)

Batch Normalisation (2)

- Calculeaza media si varianta pe fiecare mini-batch.
- Centreaza si re-scaleaza outputul unui layer (pregatirea inputului pentru urmatorul layer).
- Are parametrii de scala si translatie “antrenabili” (se invata prin procesul de training in asa fel incat poate sa pastreze expresivitatea unor caracteristici, daca este cazul)

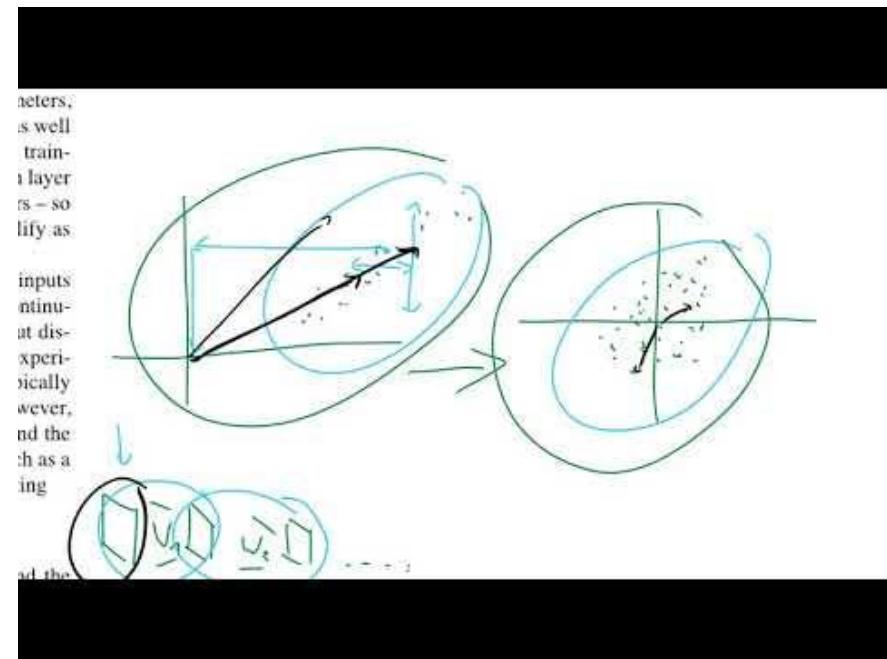
OBS :

- Daca inlocuim alpha cu $stdev(x)$ si beta cu $avg(x)$ obtinem :
 $BN(x) = x$
- Oferim retelei neurale sansa de a lasa setul de date neafectat,
daca aceasta este cea mai buna solutie pentru reprezentarea lor.

$$\hat{x} = \frac{x - avg_{batch}(x)}{stdev_{batch}(x) + \epsilon}$$

$$BN(x) = \alpha\hat{x} + \beta$$

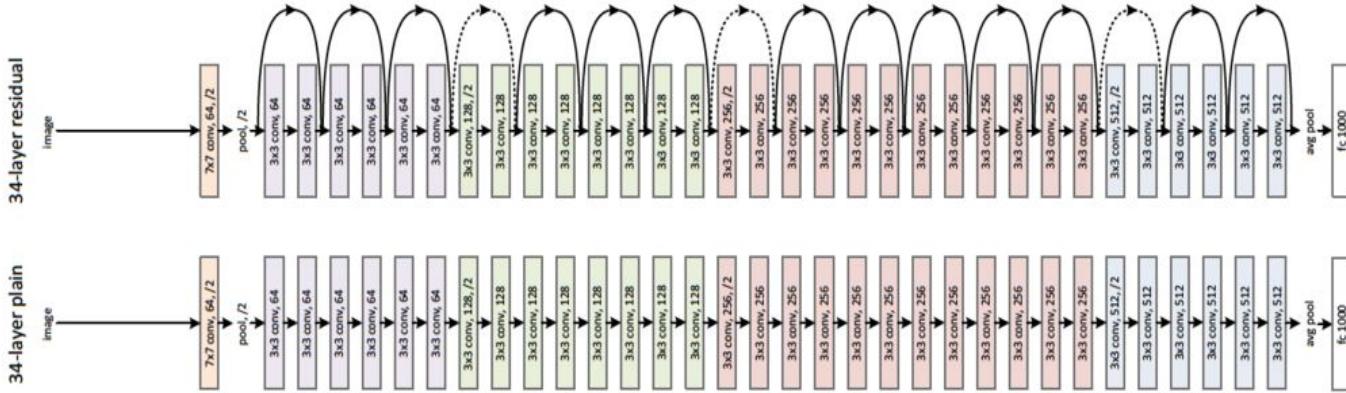
Materiale Auxiliare BatchNorm



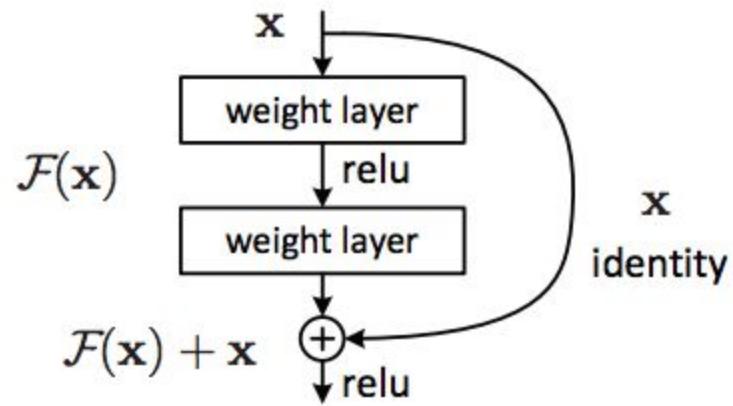
ResNet (2015)

- 3.56% eroare de clasificare top 5 pe ImageNet
- **Problema:**
 - predispunere la overfitting
- Problema retelelor neurale adanci: diminuarea gradientilor.
- Intuitie: putem adauga Identity ($f(x) = x$) de un număr nelimitat de ori fără a compromite acuratețea retelei
- **Solutie:**
 - adăugarea de conexiuni *reziduale*
 - retele neurale mai adanci
 - adăugarea unor funcții de loss suplimentare în straturile intermediare (vezi GoogLeNet)

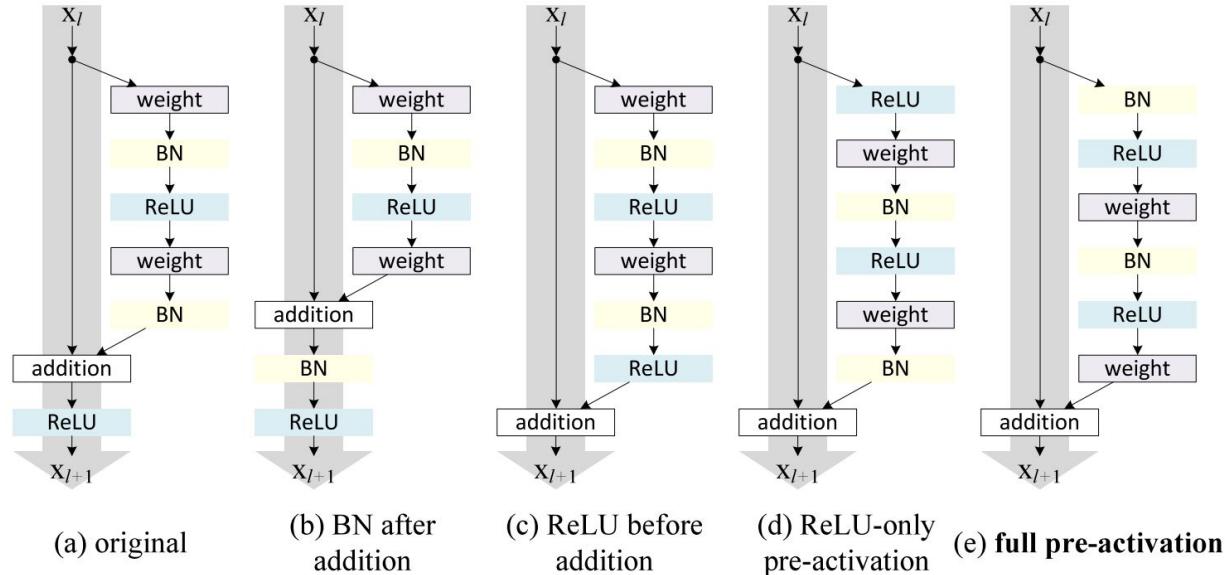
ResNet - Arhitectură



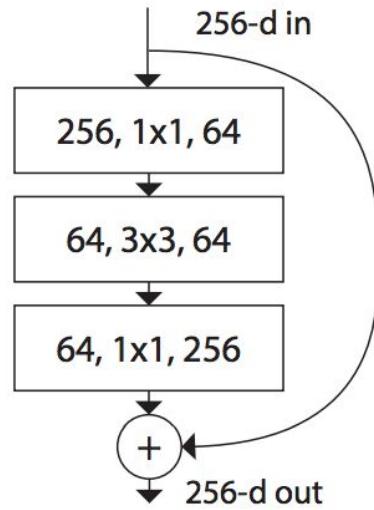
Blocul Rezidual - Arhitectură



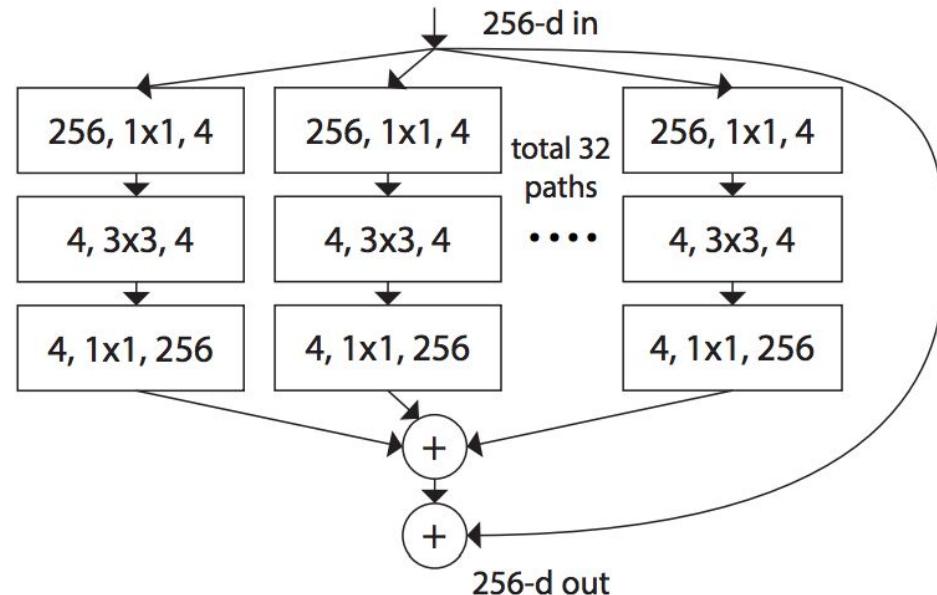
Blocul Rezidual - Variante



ResNeXt (2016)

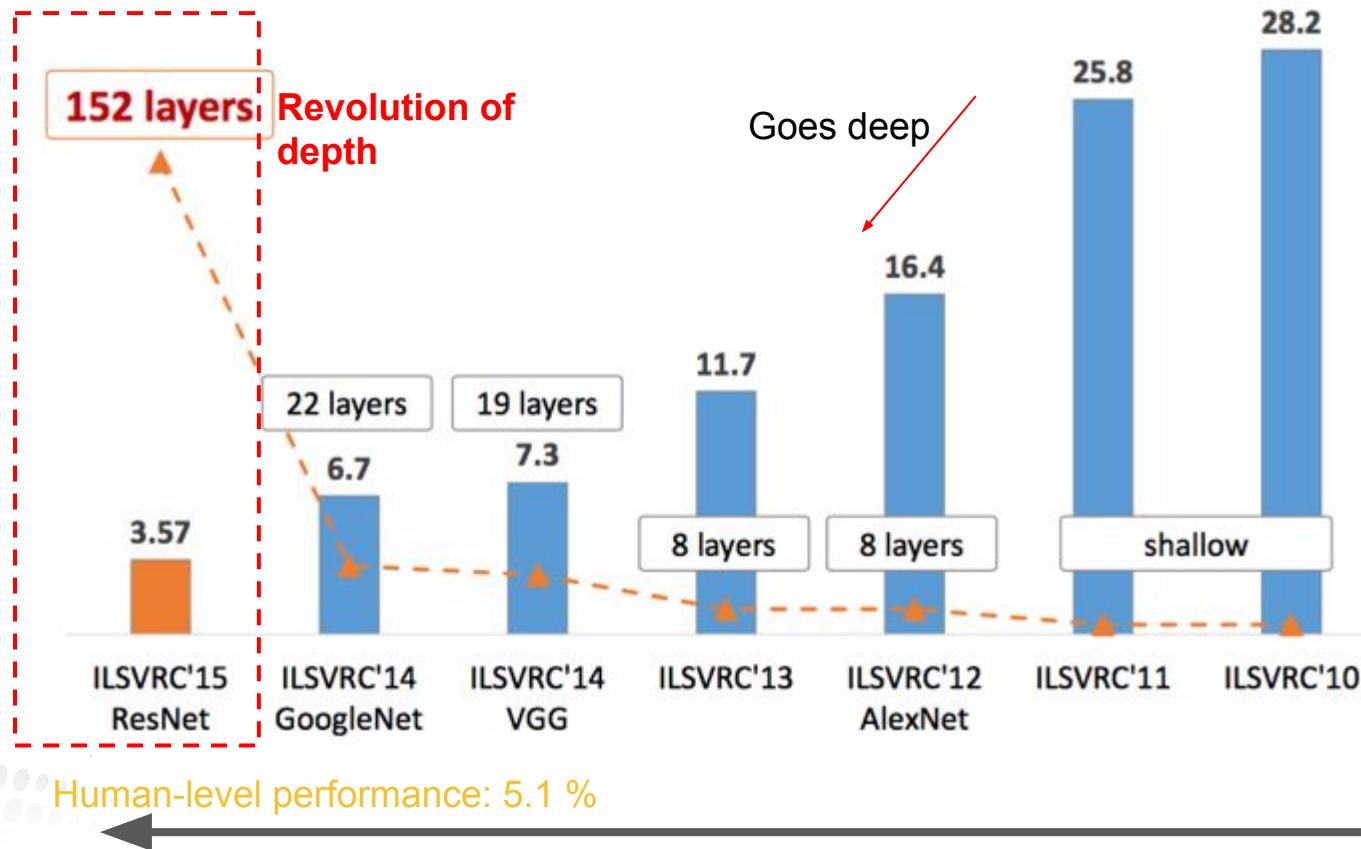


ResNet



ResNeXt

Castigatorii ILSVRC



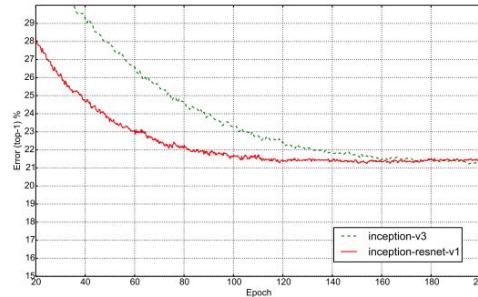
Inception-ResNet (2016)

- 3.08% eroare de clasificare top-5 pe ImageNet
- Se pune întrebarea dacă există vreun beneficiu în adăugarea de *conexiuni reziduale* în arhitectura *Inception*



Inception-ResNet (2016)

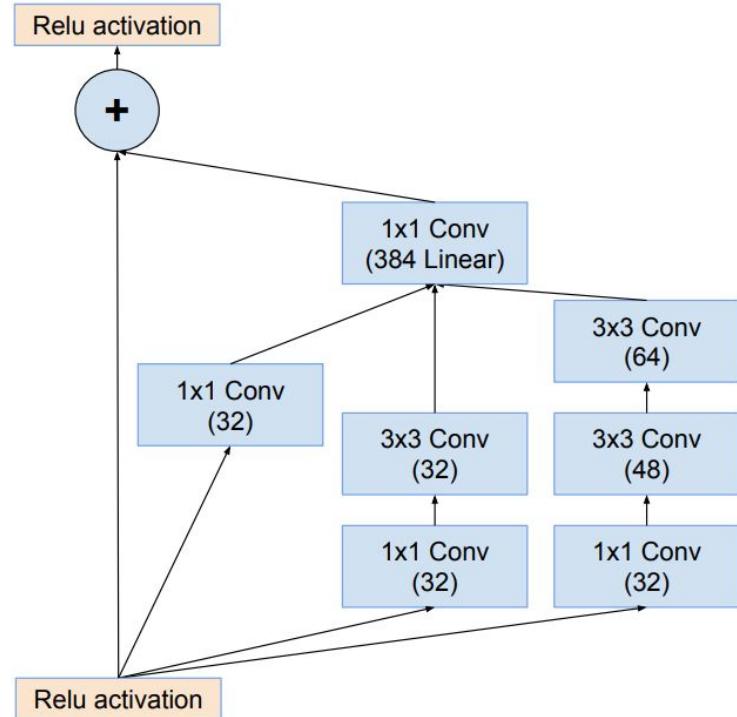
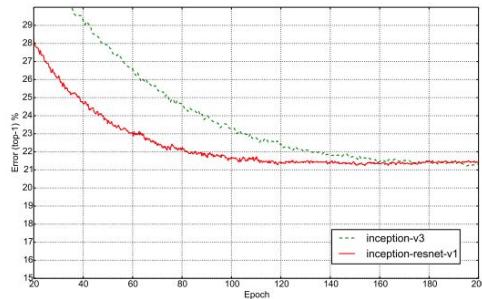
- 3.08% eroare de clasificare top-5 pe ImageNet
- Se pune întrebarea dacă există vreun beneficiu în adăugarea de *conexiuni reziduale* în arhitectura *Inception*
- Conexiunile reziduale accelerează semnificativ antrenarea arhitecturilor Inception



[Inception-V4, Inception-ResNet and the Impact of Residual Connections on Learning](#)

Inception-ResNet (2016)

- 3.08% eroare de clasificare top-5 pe ImageNet
- Se pune întrebarea dacă există vreun beneficiu în adăugarea de *conexiuni reziduale* în arhitectura *Inception*
- Conexiunile reziduale accelerează semnificativ antrenarea arhitecturilor Inception

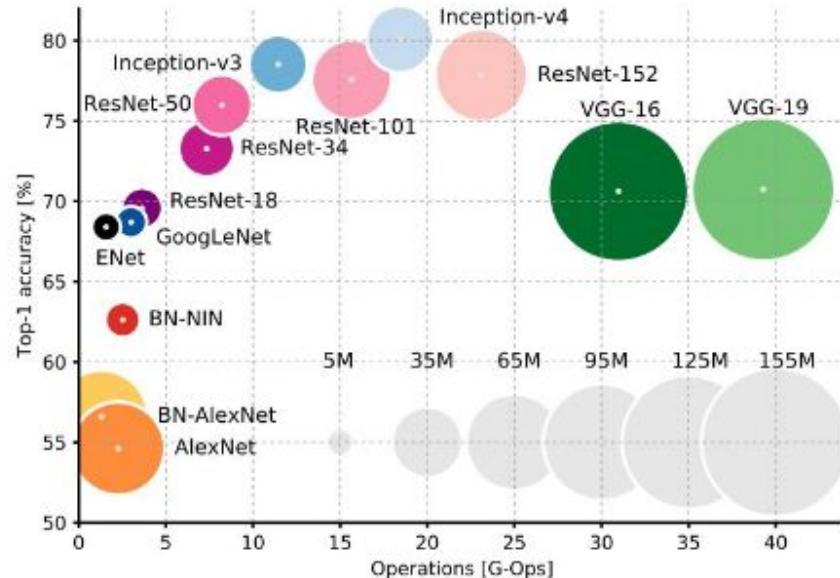
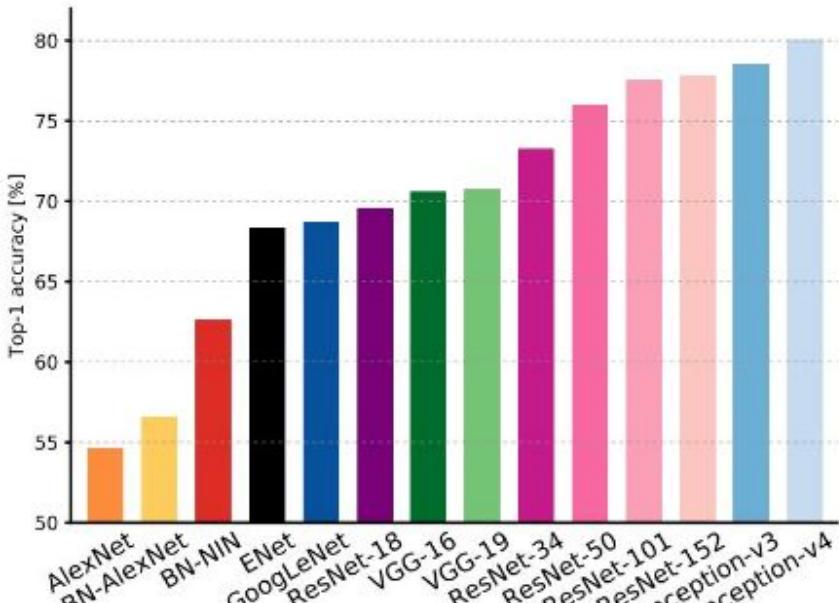


NASNet (2017)

- Imbunatatiri ale acuratetii:
 - 3.8% eroare de clasificare top-5 pe ImageNet
 - 17.3% eroare de clasificare top-1 (cu 1.2% mai bine decat cea mai buna retea)
- Provocare: Automatizarea procesului de proiectare a retelelor convolutionale prin *invataarea* de arhitecturi direct pe setul de date de interes (Google AutoML)
- **Problema:**
 - Gasirea unei arhitecturi bune pe ImageNet - practic imposibila datorita dimensiunii uriasa a setului.
- **Solutie:**
 - Cautarea unei retele bune (folosind tehnici precum RL) pe un set mult mai mic (CIFAR-10) si transferul acesteia pe ImageNet.
- Pentru a fi scalabila, complexitatea arhitecturii trebuie sa fie independenta de adancimea retelei si de dimensiunea inputului => utilizarea *celulelor* convolutionale cu structura identica, dar parametri diferiti.

Evolution of Depth

Nota: raza cercului: memory footprint



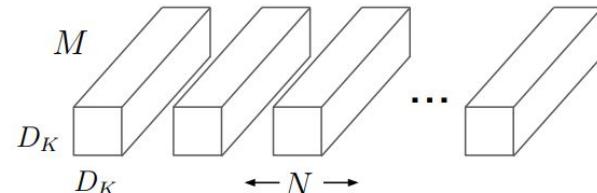
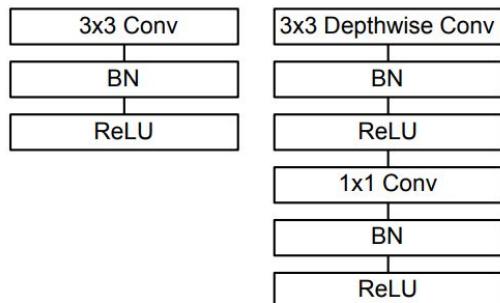
Evolution of Depth (2)

Year	CNN	Developed by	Place	Top-5 error rate	No. of parameters
1998	LeNet(8)	Yann LeCun et al			60 thousand
2012	AlexNet(7)	Alex Krizhevsky, Geoffrey Hinton, Ilya Sutskever	1st	15.3%	60 million
2013	ZFNet()	Matthew Zeiler and Rob Fergus	1st	14.8%	
2014	GoogLeNet(19)	Google	1st	6.67%	4 million
2014	VGG Net(16)	Simonyan, Zisserman	2nd	7.3%	138 million
2015	ResNet(152)	Kaiming He	1st	3.6%	

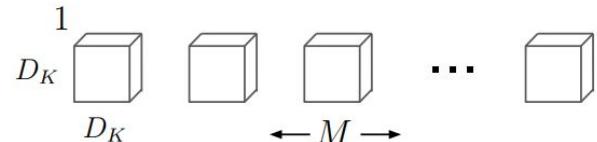
MobileNets [Google, 2017]

Aproximeaza convolutia cu 2 operatii:

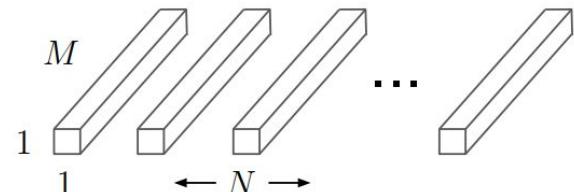
- $D_K \times D_K \times 1$: convolutie **depthwise** separata pe canale - fiecare filtru vede un singur canal de input
- $1 \times 1 \times M$: convolutie **pointwise** - fiecare filtru se extinde in toata adancimea input-ului.
- Reduce numarul de parametri:
 - Separabil: $N \times (D_K \times D_K + M)$ parametri
 - Standard: $N \times (D_K \times D_K \times M)$ parametri



(a) Standard Convolution Filters

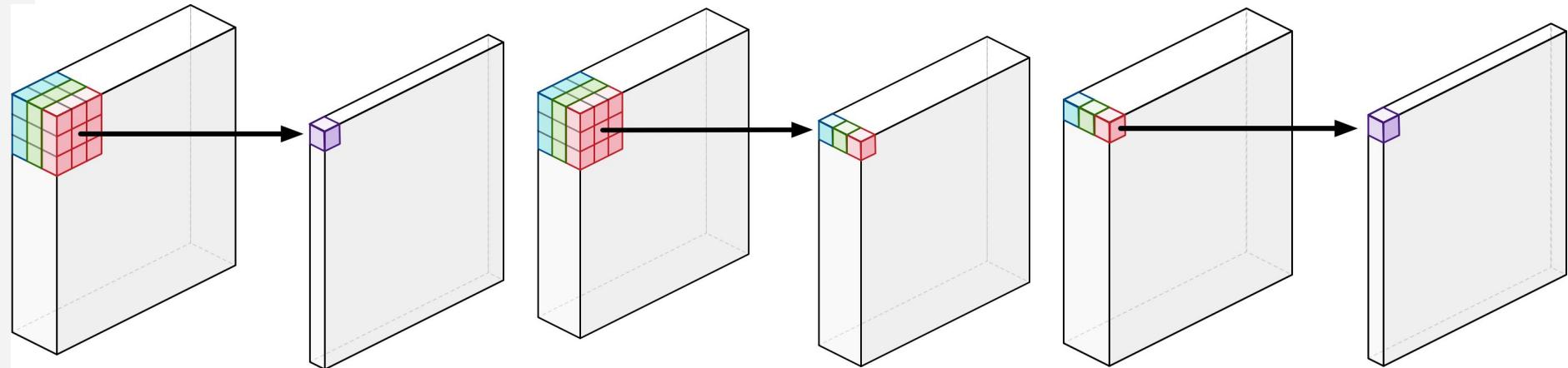


(b) Depthwise Convolutional Filters



(c) 1×1 Convolutional Filters called Pointwise Convolution in the context of Depthwise Separable Convolution

Convolutii Depthwise separabile



Convolutia normala ($K \times K \times C$)

- full depth in input
- kernel size in spatiu (h,w)

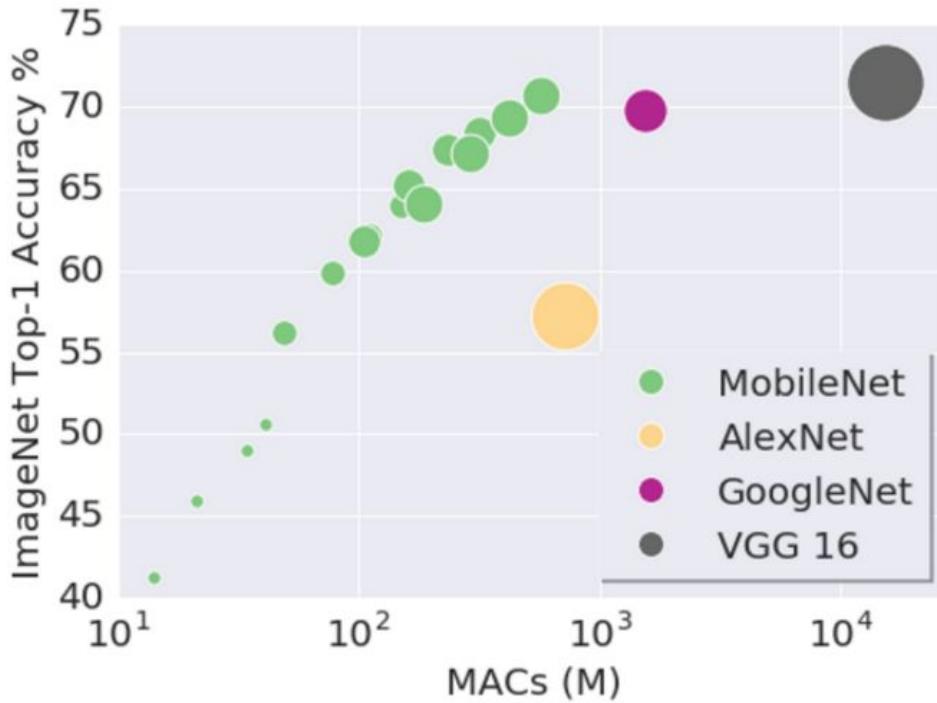
Convolutia depthwise ($K \times K \times 1$)

- pe un singur canal in input
- Kernel size in spatiu (h,w)

Convolutia pointwise ($1 \times 1 \times C$)

- full depth in input
- 1×1 in spatiu (h,w)

Convolutii Depthwise separabile



Acuratete ILSVRC:

- MobileNets cu input 224x224 si full depth Top-5 89.9 %
- **Top-1 : 70.9 %**

Nota:

- GoogleNet: 69 % Top-1
- **VGG-16: 68.5 % Top-1**

MobileNets:

- 4.2M parametri
- 569M Operatii Multiply-Add (224x224)

VGG-16 Net:

- 138M parametri
- 15.5G operatii Multiply-add (224x224)