Large Scale Filter: How can we realize the fully connection as convolution filter? Let the filter size is the same as image size.

VERY DEEP CONVOLUTIONAL NETWORKS FOR LARGE-SCALE IMAGE RECOGNITION

ICLR
(International Conference on Learning Representations)
2015

Karen Simonyan* & Andrew Zisserman+
Visual Geometry Group, Department of Engineering Science, University of Oxford
{karen,az}@robots.ox.ac.uk

Student: 林超, Edward, Gimmy
Advisor: Jenn-Jier James Lien (連震杰)
Robotics Lab CSIE NCKU

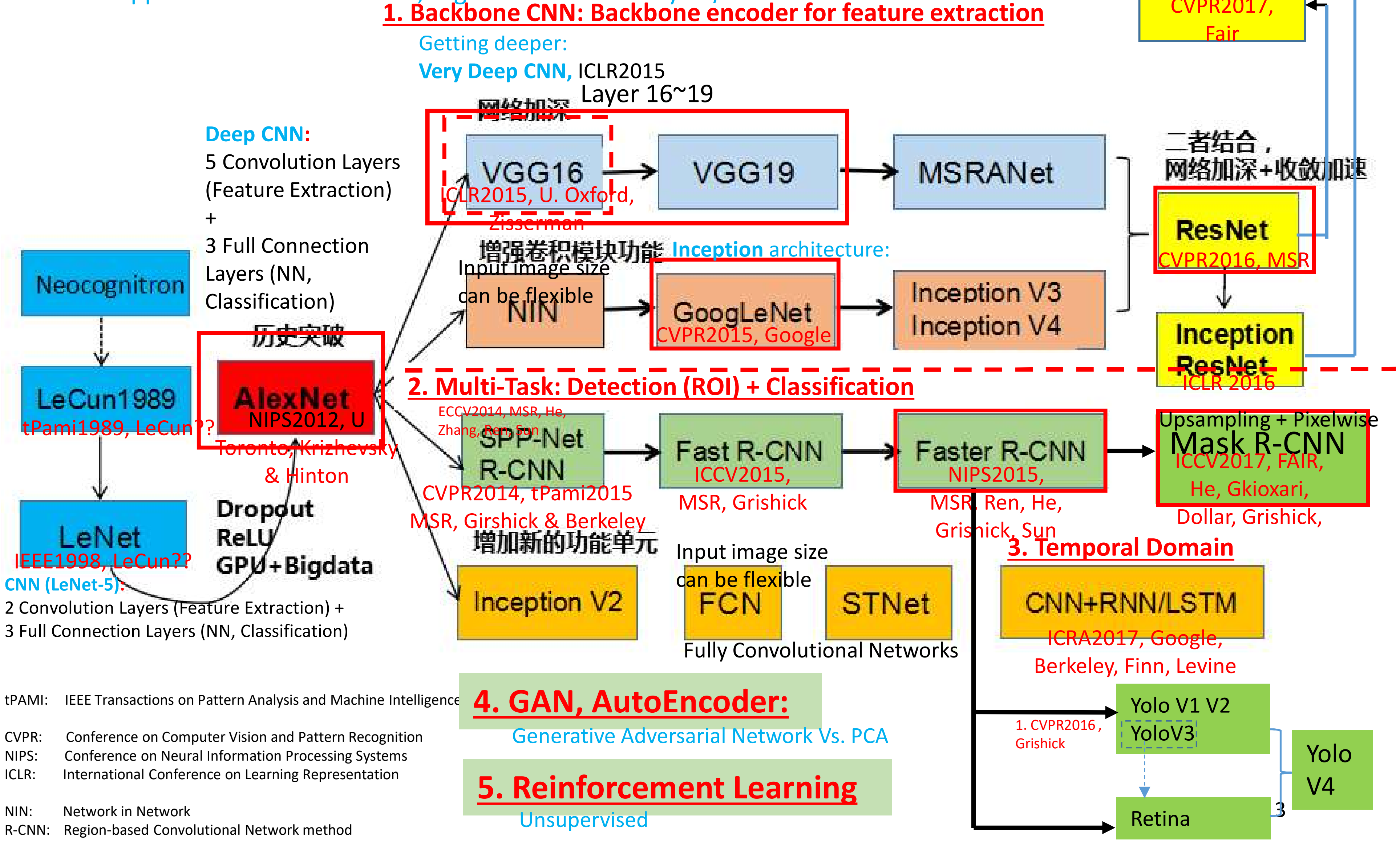
Outline

0. Abstract
1. Introduction
2. Convnet configurations
3. Classification framework
4. Classification experiments
5. Conclusion

0.1 Deep Learning: History

Which one do I want to be?

1. Basic CNN Creation
2. Modified CNN
3. CNN Application: Data collection, organization and analysis / benchmark



tPAMI: IEEE Transactions on Pattern Analysis and Machine Intelligence

CVPR: Conference on Computer Vision and Pattern Recognition
NIPS: Conference on Neural Information Processing Systems
ICLR: International Conference on Learning Representation

NIN: Network in Network
R-CNN: Region-based Convolutional Network method
SPP-Net: Spatial Pyramid Pooling networks

0. Abstract

Main contribution:

- thorough evaluation of networks of increasing depth using an architecture with very small (3×3) convolution filters.

- A significant improvement on the prior-art configurations can be achieved by pushing the depth to 16–19 weight layers.

- Our team secured the first and the second places (of ImageNet) in the localization and classification tracks respectively. We also show that our representations generalize well to other datasets.

1. Introduction (1/2)

1) ConvNets have recently enjoyed a great success in large-scale image and video recognition (Krizhevsky et al. 2012; Zeiler & Fergus 2013; Sermanet et al; 2014; Simonyan & Zisserman 2014) which has become possible due to the large public image repositories such as ImageNet.
2) A number of attempts have been made to improve the original architecture of Krizhevsky et al. (2012 AlexNet) in a bid to achieve better accuracy.
   - The best-performing submissions to the ILSVRC-2013 (Zeiler & Fergus 2013; Sermanet et al. 2014) utilized smaller receptive window size and smaller stride of the first convolutional layer.
   - Training and testing the networks densely over the whole image and over multiple scales (Sermanet et al.2014; Howard 2014).
   - We address another important aspect of ConvNet architecture design – its depth. we fix other parameters of the architecture and steadily increase the depth of the network by adding more convolutional layers.
3) As a result we come up with significantly more accurate ConvNet architectures which not only achieve the state-of-the-art accuracy on ILSVRC classification and localisation tasks but are also applicable to other image recognition datasets.
4) We have released our two best-performing models1 to facilitate further research

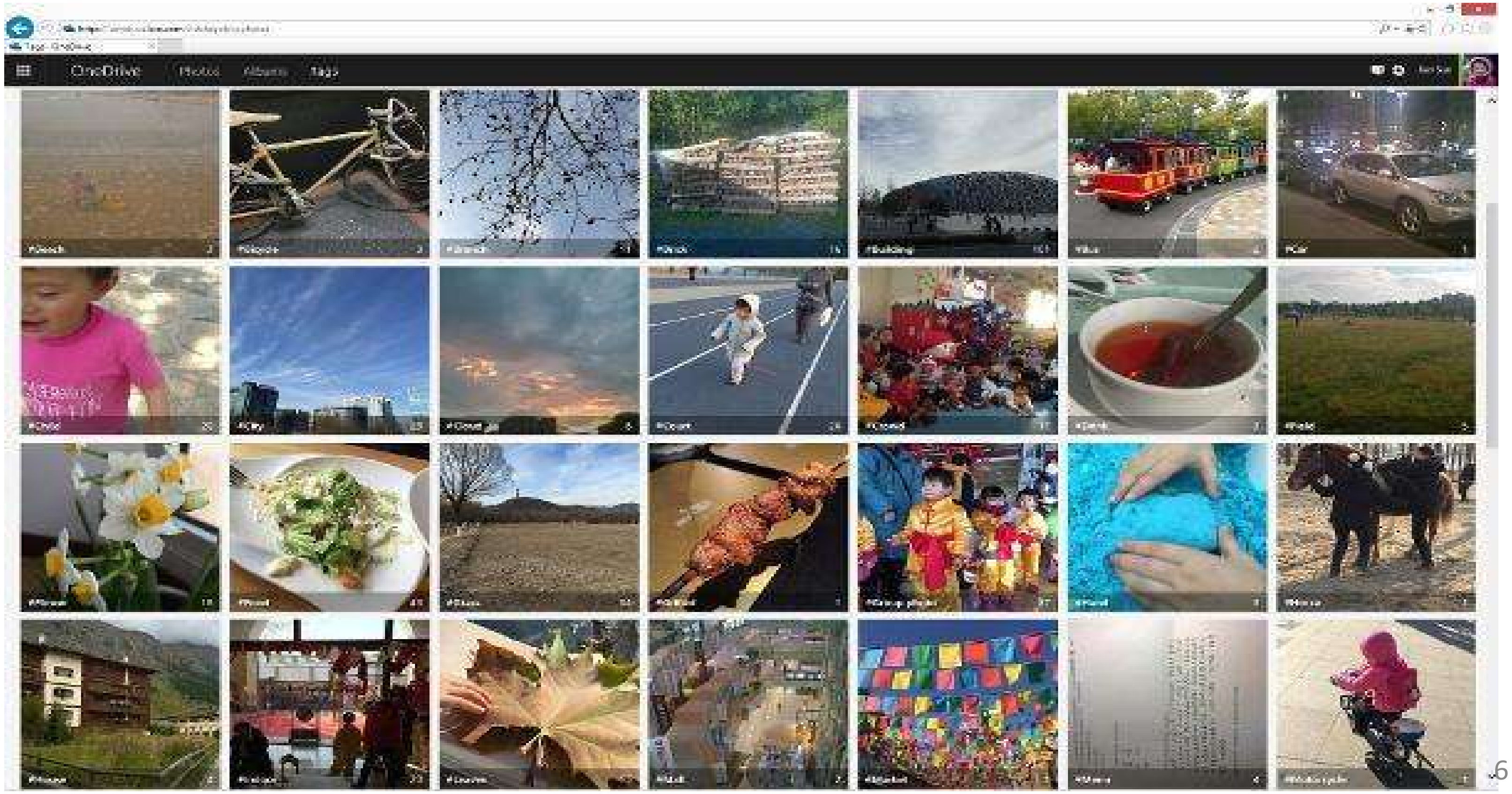
1. Introduction (1/2)

Dataset:

1) ImageNet is a dataset of over 15 million labeled high-resolution images belonging to roughly 22000 categories.

2) ILSVRC uses a subset of ImageNet with roughly 1000 images in each of 1000 categories. In all there are roughly 1.3 million training images 50,000 validation images and 100,000 testing images.

1/26 1/13

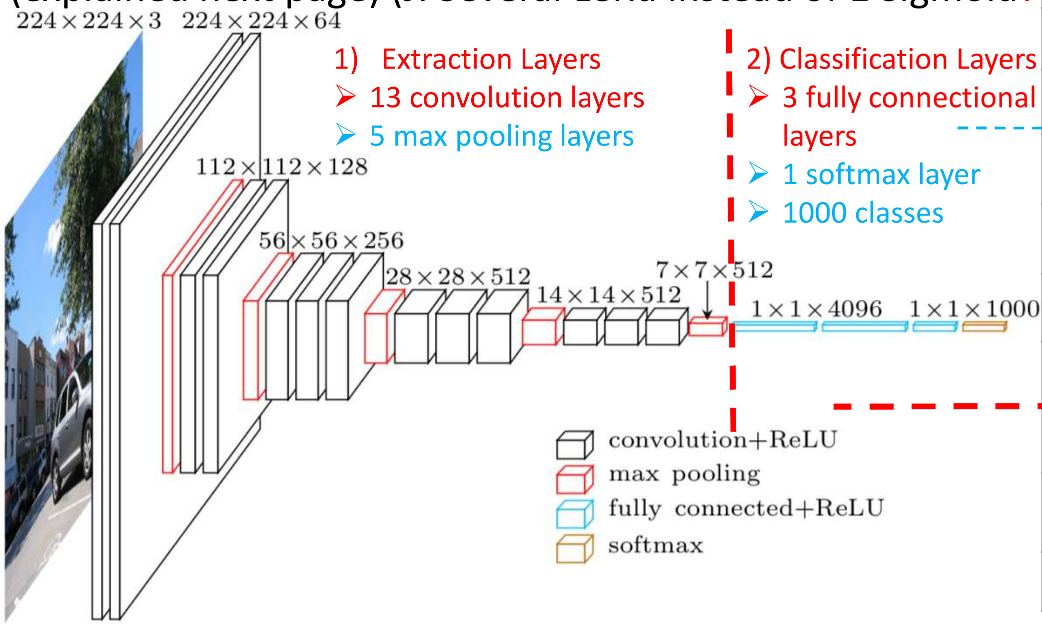


2. VGG16 Configurations (1/2)

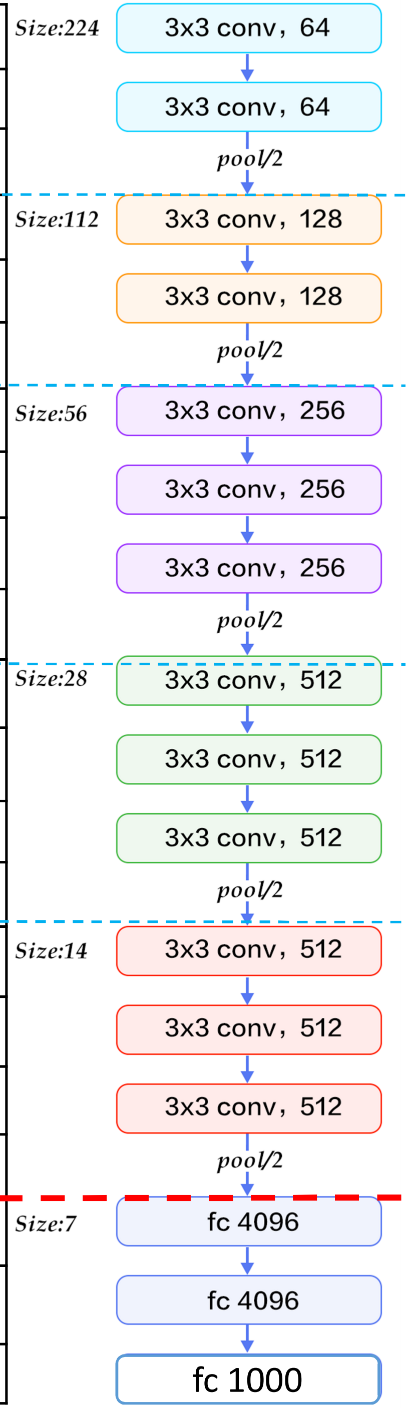
- Spatial filter size: 3×3
- Stride: 1
- Padding: 1
- Non-overlap pooling
- Total parameters:

$$params = weights + biases\ (in\ FC)$$
$$= 138344128 + 4096 * 3 = 138{,}356{,}416$$

→ Instead of using large spatial filter size (7×7), use small one 3×3
→ reduce parameters, and increase the net depth (or # of ReLU)
→ make the decision boundary more discriminative (explained next page) (J: Several LeRu instead of 1 Sigmoid??)



weights
(3*3*3)*64 = 1728
(3*3*64)*64 = 36864
0
(3*3*64)*128 = 73728
(3*3*128)*128 = 147456
0
(3*3*128)*256 = 294912
(3*3*256)*256 = 589824
(3*3*256)*256 = 589824
0
(3*3*256)*512 = 1179648
(3*3*512)*512 = 2359296
(3*3*512)*512 = 2359296
0
(3*3*512)*512 = 2359296
(3*3*512)*512 = 2359296
(3*3*512)*512 = 2359296
0
7*7*512*4096 = 102,760,448
4096*4096 = 16,777,216
4096*1000 = 4096000



2. VGG16 Configurations (2/2)

3: 3x3 filter convolution
64: Feature maps

16 and 19 have better results

Subsampling by 2 for each max-pooling

ConvNet Configuration					
A	A-LRN	B	C	D	E
11 weight layers	11 weight layers	13 weight layers	16 weight layers	16 weight layers	19 weight layers
input (224 × 224 RGB image)					
conv3-64	conv3-64 **LRN**	conv3-64 **conv3-64**	conv3-64 conv3-64	conv3-64 conv3-64	conv3-64 conv3-64
maxpool					
conv3-128	conv3-128	conv3-128 **conv3-128**	conv3-128 conv3-128	conv3-128 conv3-128	conv3-128 conv3-128
maxpool					
conv3-256 conv3-256	conv3-256 conv3-256	conv3-256 conv3-256	conv3-256 conv3-256 **conv1-256**	conv3-256 conv3-256 **conv3-256**	conv3-256 conv3-256 conv3-256 **conv3-256**
maxpool					
conv3-512 conv3-512	conv3-512 conv3-512	conv3-512 conv3-512	conv3-512 conv3-512 **conv1-512**	conv3-512 conv3-512 **conv3-512**	conv3-512 conv3-512 conv3-512 **conv3-512**
maxpool					
conv3-512 conv3-512	conv3-512 conv3-512	conv3-512 conv3-512	conv3-512 conv3-512 **conv1-512**	conv3-512 conv3-512 **conv3-512**	conv3-512 conv3-512 conv3-512 **conv3-512**
maxpool					
FC-4096					
FC-4096					
FC-1000					
soft-max					

Table 1: **ConvNet configurations** (shown in columns). The depth of the configurations increases from the left (A) to the right (E), as more layers are added (the added layers are shown in bold). The convolutional layer parameters are denoted as "conv⟨receptive field size⟩-⟨number of channels⟩". The ReLU activation function is not shown for brevity.

2. VGG16 Configurations: 3x3 Filter Instead of 7x7 filter (1/3)

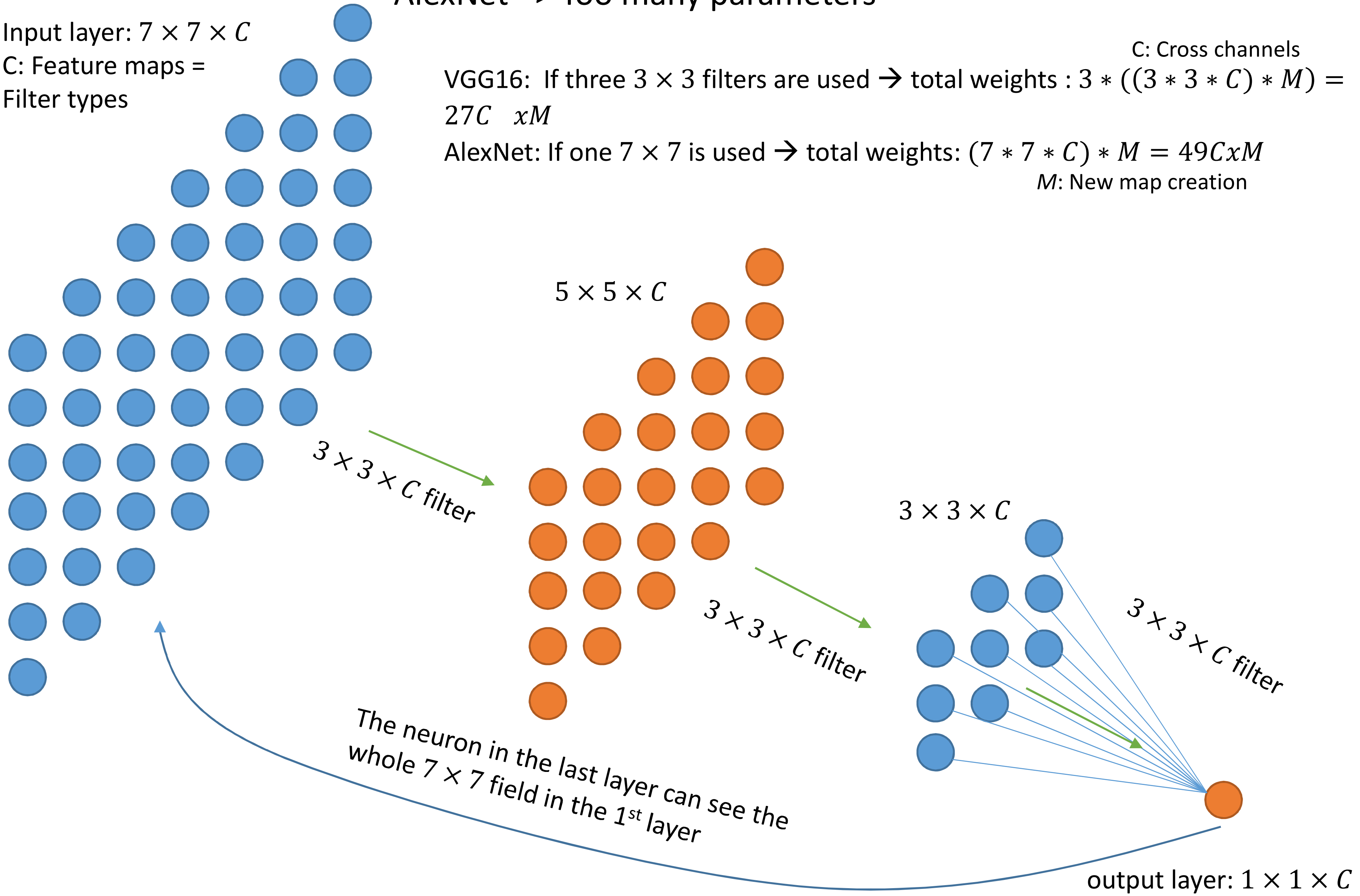
AlexNet => Too many parameters

C: Cross channels

VGG16: If three 3×3 filters are used → total weights : $3 * ((3 * 3 * C) * M) = 27C \ \ xM$

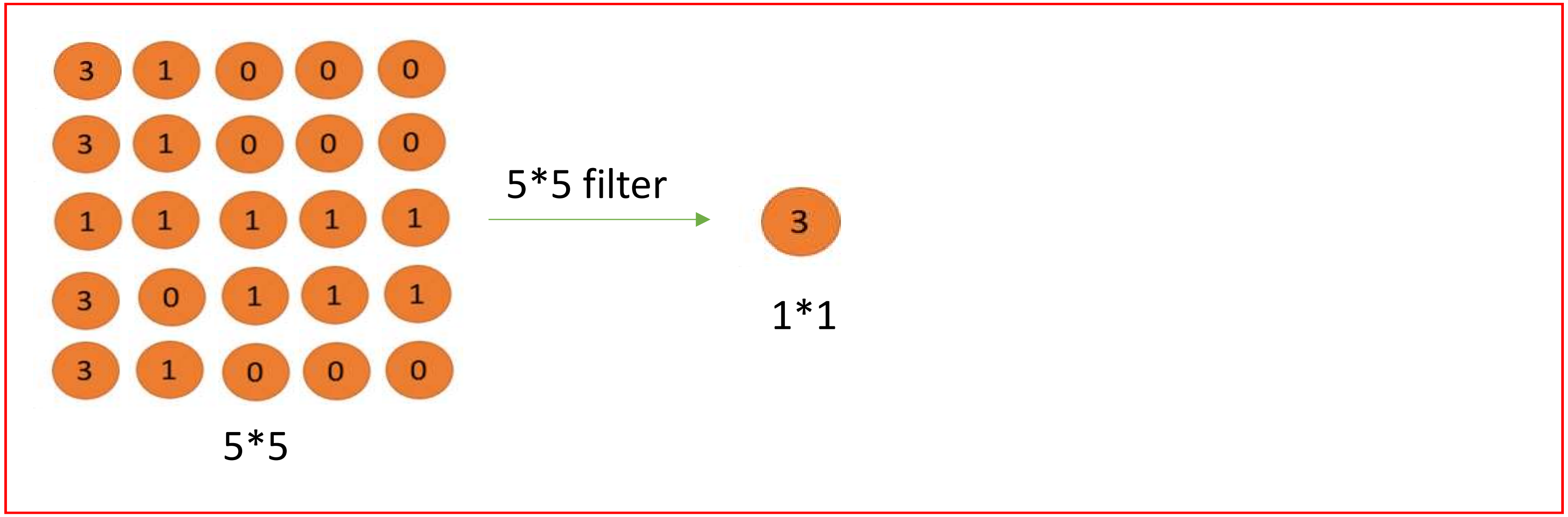
AlexNet: If one 7×7 is used → total weights: $(7 * 7 * C) * M = 49CxM$

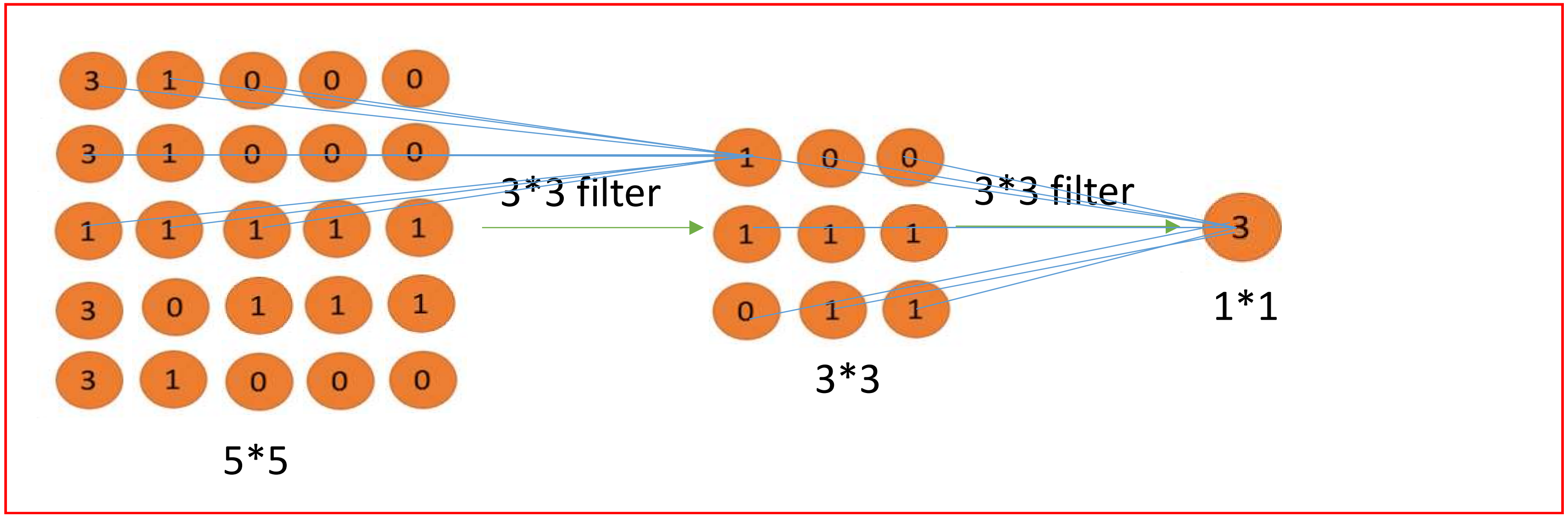
M: New map creation



2. VGG16 Configurations: 3x3 Filter Instead of 5x5 filter (2/3)

Example X:





2. VGG16 Configurations: 3x3 Filter Instead of 7x7 filter (3/3)

Discussion

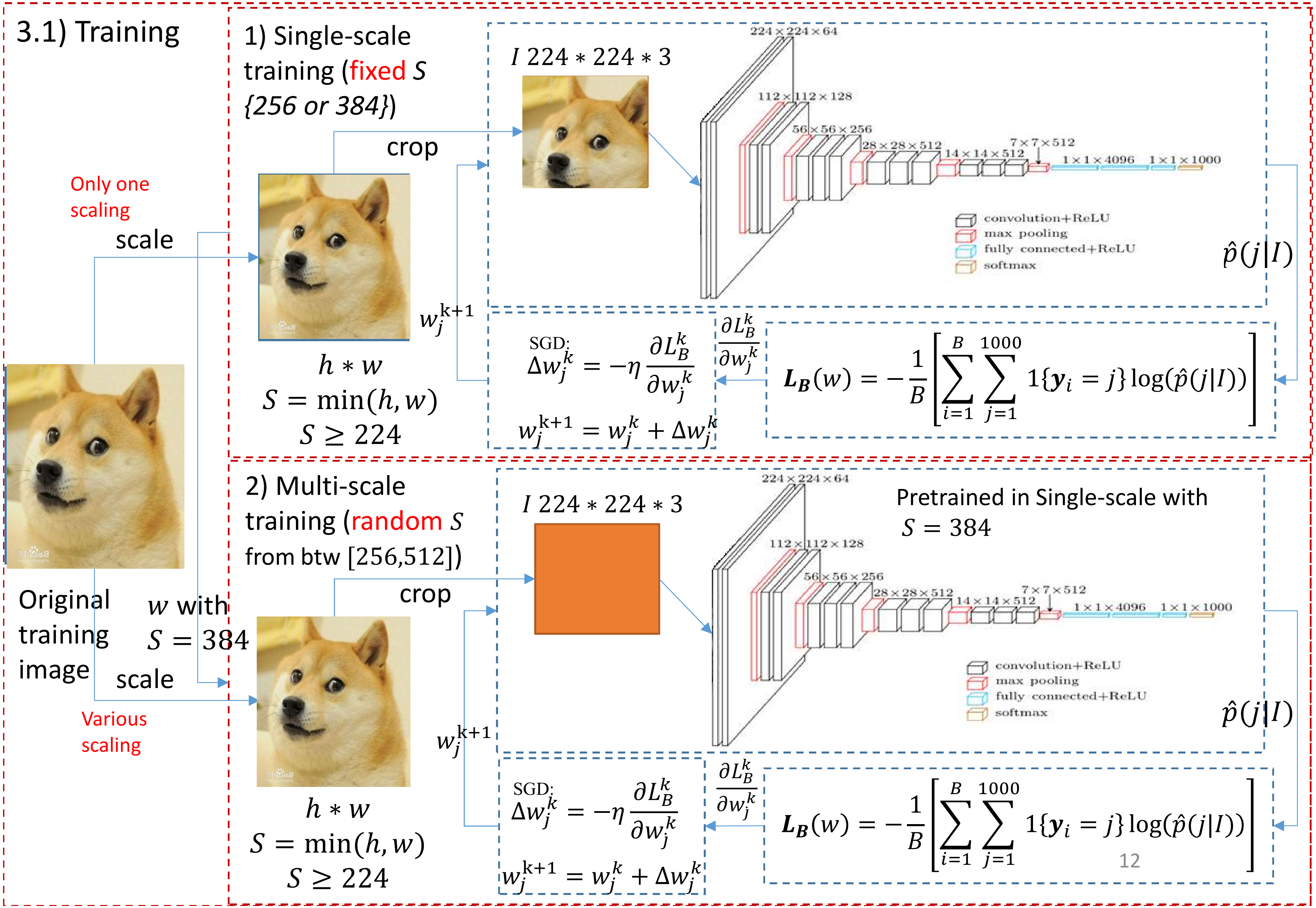
- Using 3*3 filter: Our ConvNet configurations are quite different from the ones used in the top-performing entries of the ILSVRC Rather than using relatively large receptive fields in the first conv. we use very small 3 × 3 receptive fields throughout the whole net.

Improve by SPP (Spatial Pyramid Pooling

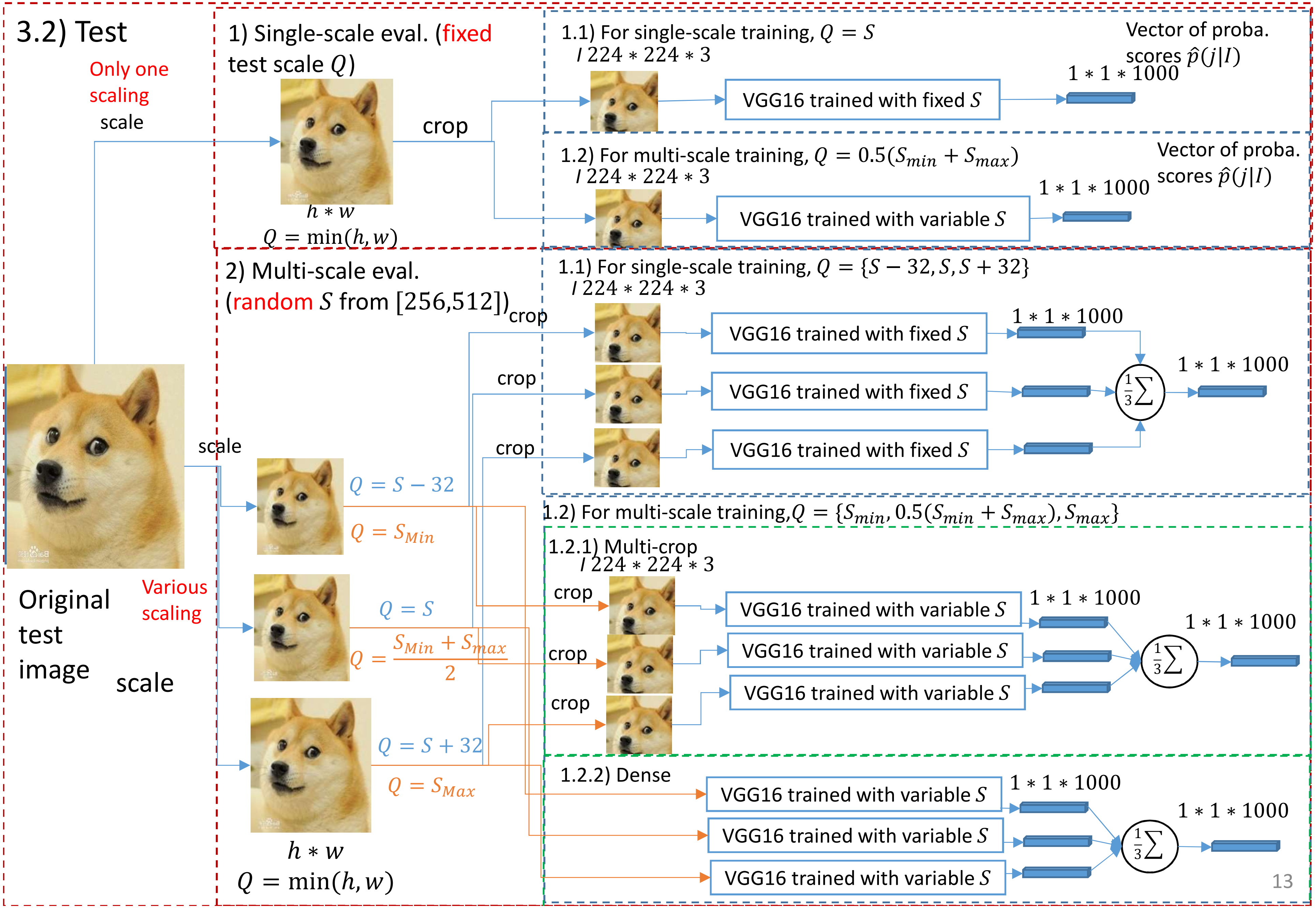
  - How: It is easy to see that a stack of two 3x3 conv. layers (without spatial pooling in between) has an effective receptive field of 5x5; three such layers have a 7 × 7 effective receptive field. (see example X)
  - Why:
    - we incorporate three non-linear rectification layers instead of a single one which makes the decision function more discriminative.
    - decrease the number of parameters 7 × 7 conv. layer would require $7^2C^2 = 49C^2$ parameters. 3 × 3 conv. layer would require $27C^2$ parameters.(C: channels)

3. Classification Framework: Training Process

Here, the purpose is to understand whether input size will affect the performance?



3. Classification Framework: Test Process



3. Classification framework (Notes) (3/12)

1) Training

Unbiasing: $I = I - \bar{I}$

Batch size: $B = 256$

Momentum: 0.9

L_2 regularization: $\lambda = 5 \times 10^{-4}$

Init. learning rate: $\eta = 10^{-2}$

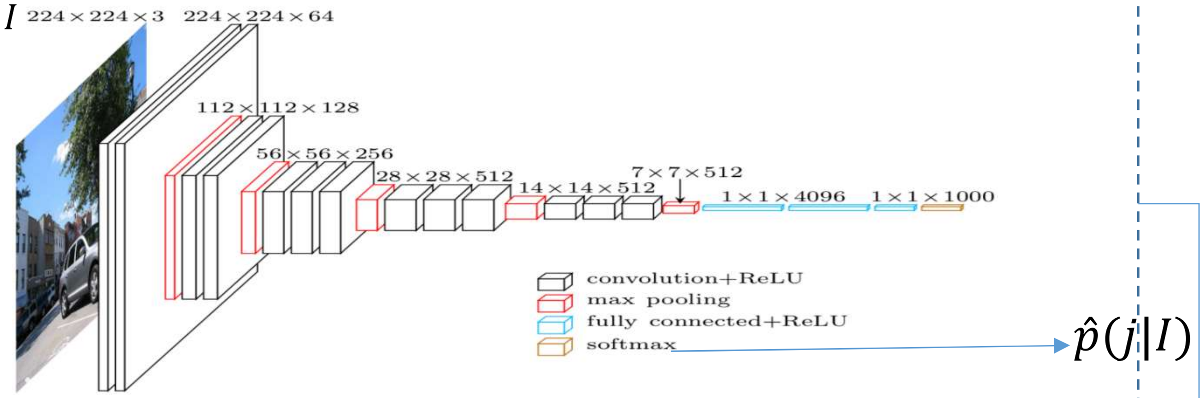
If validation accuracy stopped improving, $\eta = 0.1 \times \eta$

Training: 74 epochs

Dropout ratio: 0.5

Xavier initialization:

$$w \sim N\left(0, \sqrt{\frac{2}{n_{in} + n_{out}}}\right)$$



$$\boldsymbol{L_B}(w) = -\frac{1}{B}\left[\sum_{i=1}^{B}\sum_{j=1}^{1000} 1\{\boldsymbol{y}_i = j\}\log(\hat{p}(j|I))\right]$$

$$\frac{\partial L_B^k}{\partial w_j^k}$$

SGD:

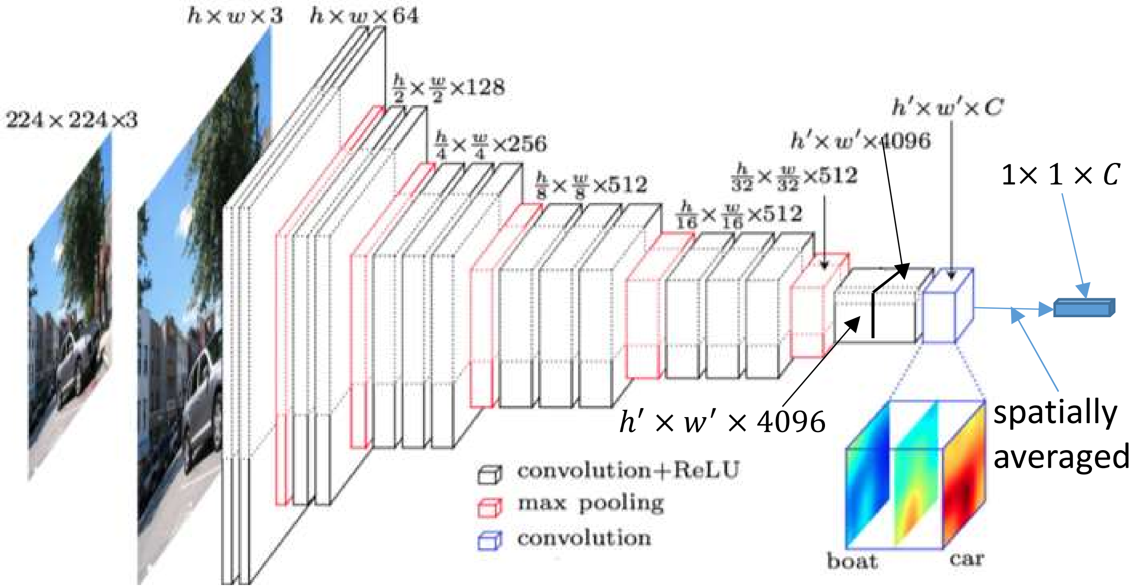
$$\Delta w_j^k = -\eta \frac{\partial L_B^k}{\partial w_j^k}$$

$$w_j^{k+1} = w_j^k + \Delta w_j^k$$

w_j^{k+1}

2) Testing

- Use trained Net in the training phase
- Spatial input image size, $h \times w$ can be larger than 224×224
- FC layers → fully convolutional layer: output is $C = 1000$ heatmaps
- The final scores obtained by averaging the scores of the scaled test image and its horizontal flip



3. Classification framework (4/12)

3.1 Training

- ConvNet training procedure: (generally follows Krizhevsky et al. (2012)).
  - batch size: 256
  - momentum: 0.9
  - dropout ratio: 0.5
  - learning rate: 10^{-2} (decreased by a factor of 10 when the validation set accuracy stopped improving)

- Training times: the learning rate was decreased 3 times and the learning was stopped after 370K iterations (74 epochs).

- converge quickly: We conjecture that in spite of the larger number of parameters and the greater depth of our nets compared to (Krizhevsky et al. 2012) the nets required less epochs to converge due to
  (a) implicit regularisation imposed by greater depth and smaller conv. filter sizes;
  (b) **pre-initialisation of certain layers.**

3. Classification framework (5/12)

Data augmentation:
Our neural network architecture has 138 million parameters this turns out to be insufficient to learn so many parameters without considerable overfitting.

1) Cropping and flipping
   1.1) horizontal reflections
   1.2) generating image translations(cut 224*224 patches from the256*256 images)

- This increases the size of our training set by a factor of 2048($(256-224)^2$x2)
- though the resulting training examples are of course highly interdependent. Without this scheme our network suffers from substantial overfitting

3. Classification framework (6/12)

2) Color augmentation

- perform PCA on the set of RGB pixel values throughout the ImageNet training set
- This scheme reduces the top-1 error rate by over 1%.

$$I_{xy} = [I_{xy}^R, I_{xy}^G, I_{xy}^B]$$

某图像的每
一个像素

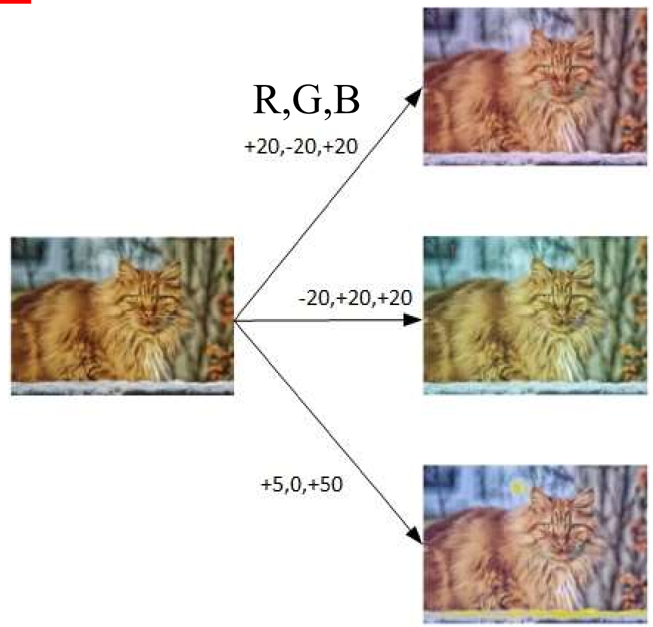
$$+$$

$$[\mathbf{p}_1, \mathbf{p}_2, \mathbf{p}_3][\alpha_1\lambda_1, \alpha_2\lambda_2, \alpha_3\lambda_3]^T$$

特征向量　特征值　隨機值（Gaussian）

物理意義：神經網絡不應該因為顏色不同而把相同的影像分到不同的類別

Pca可以找出自然圖片中差異比較大的rgb改變方向



This method will perturb the image colors along these PCA axes. If PCA vectors have larger eigenvalue than the others, so it was clearly dominant and can be equivalent with brightness perturbation instead of color perturbation.

3. Classification framework (7/12)

- The initialization:
  - we began with training the configuration A (Table 1) shallow enough to be trained with random initialization
  - when training deeper architectures we initialised the first four convolutional layers and the last three fully connected layers with the layers of net A (the intermediate layers were initialised randomly).

- random initialization: we sampled the weights from a normal distribution with the zero mean and 10^{-2} variance.

- Other way to initialization: It is worth noting that after the paper submission we found that it is possible to initialise the weights without pre-training by using the random initialisation procedure of Glorot & Bengio (2010).

- Training time: On a system equipped with four NVIDIA Titan Black GPUs training a single net took 2–3 weeks depending on the architecture.

3. Classification framework (8/12)

3.2 Training image size

- Let S be the smallest side of an rescaled training image from which the ConvNet input is cropped (224*224). We consider two approaches for setting the training scale S

  1) The first is to fix S which corresponds to single-scale training. we evaluated models trained at two fixed scales: S =256 (which has been widely used in the prior art (Krizhevsky et al. 2012) and S = 384.

  2) The second approach to setting S is multi-scale training where each training image is individually rescaled by randomly sampling S from a certain range [Smin Smax] (we used S_{min} = 256 and S_{max} = 512).

3. Classification framework (9/12)

3.3 Testing

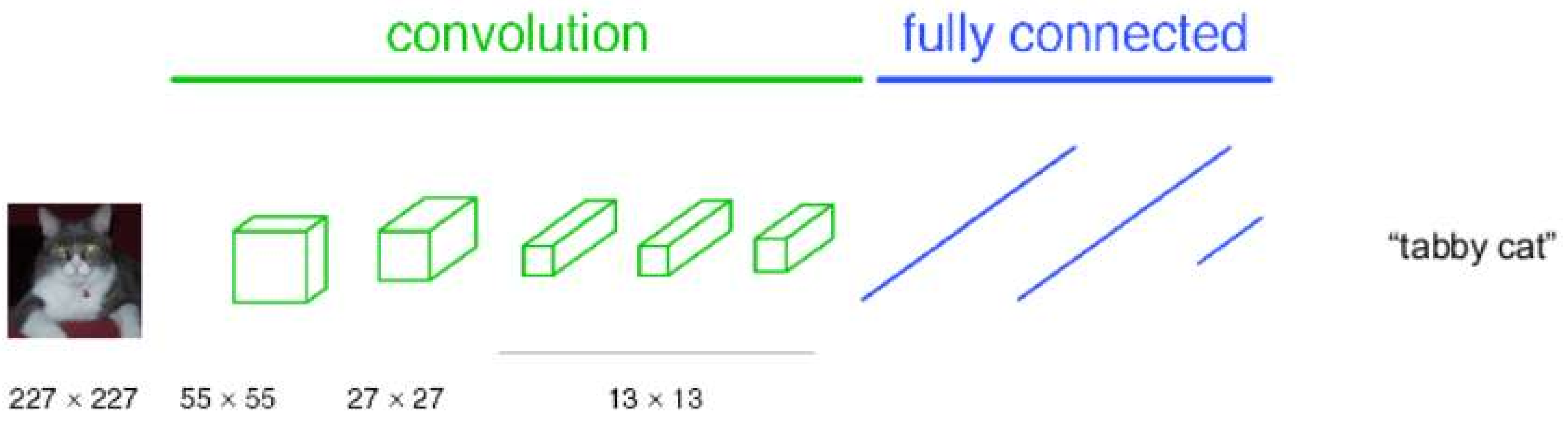
- At test time given a trained ConvNet and an input image it is classified in the following way.
  - First it is rescaled to a pre-defined smallest image side denoted as Q.

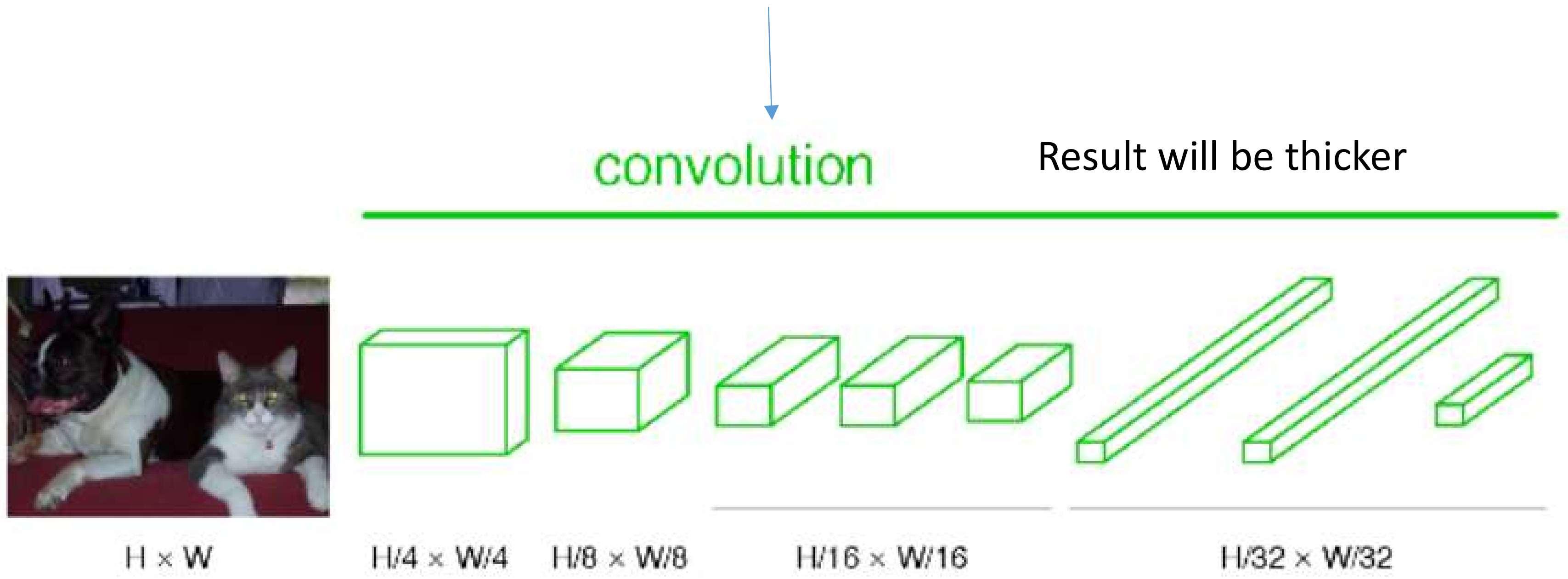
  - Then the fully-connected layers are first converted to convolutional layers (fully-convolutional) (the first FC layer to a 7×7 conv. layer the last two FC layers to 1×1 conv. layers).

  - Since the fully-convolutional network is applied over the whole image there is no need to sample multiple crops at test time.

- At the same time using a large set of crops can lead to improved accuracy as it results in a finer sampling of the input image compared to the fully-convolutional net.

3. Classification framework (10/12)





3. Classification framework (11/12)

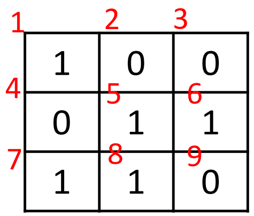
Example

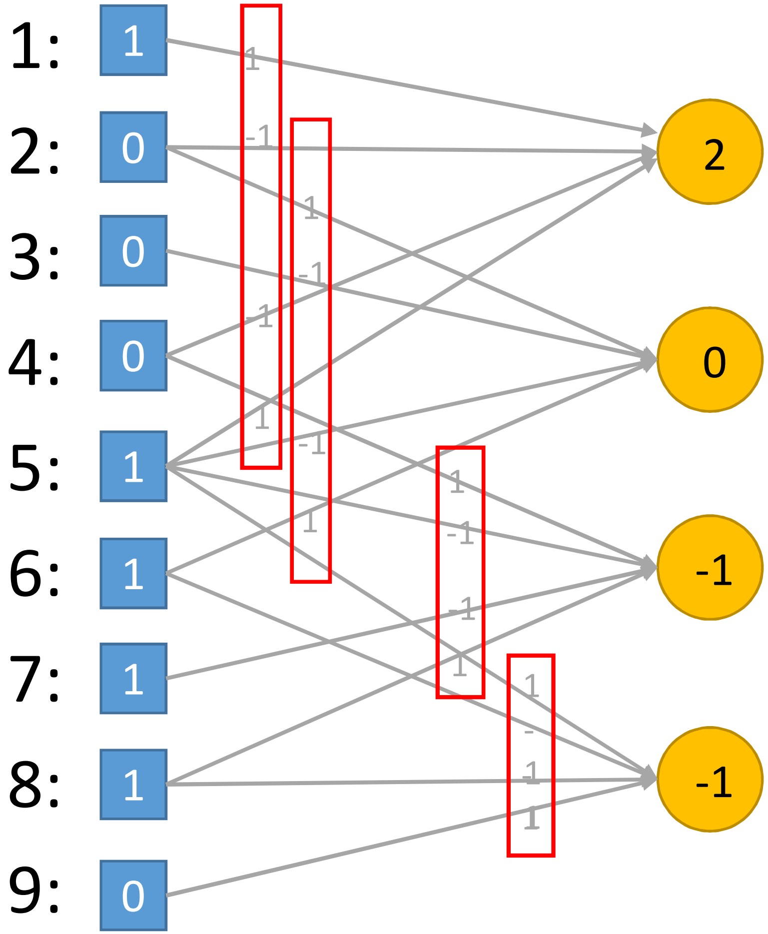
2*2 filter

1	-1
-1	1

3*3 Image

1	0	0
0	1	1
1	1	0





Shared weights

Less parameters

fully-connected		CNN
9*4	→	4

3. Classification framework (12/12)

Example

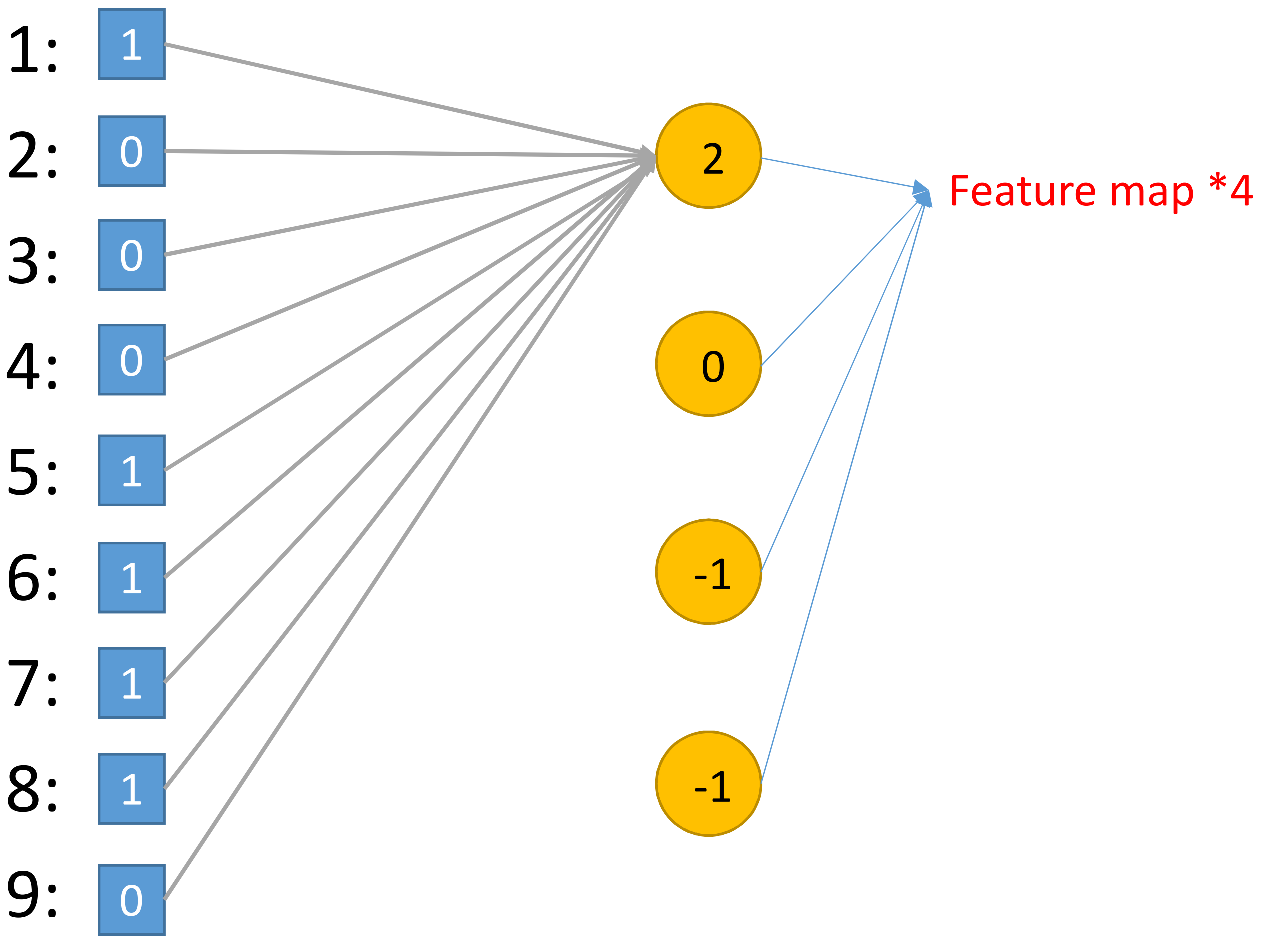
Therefore we need to design one 3*3 filter

1	1	-1
-1	-1	1
1	0	1

Suppose 3*3 Image

1	0	0
0	1	1
1	1	0

Large Scale Filter: How can we realize the fully connection as convolution filter? Let the filter size is the same as image size.



4. Classification experiment (1/4)

4.1 Single scale evaluation

Test scale $Q = S$ for models trained with fixed S, and $Q = 0.5(S_{min} + S_{max})$ for model trained with jittered $S \in [S_{min}, S_{max}]$

Table 3: **ConvNet performance at a single test scale.**

ConvNet config. (Table 1)	smallest image side		top-1 val. error (%)	top-5 val. error (%)
	train (S)	test (Q)		
A	256	256	29.6	10.4
A-LRN	256	256	29.7	10.5
B	256	256	28.7	9.9
C (Use 1*1 filter)	256	256	28.1	9.4
	384	384	28.1	9.3
	[256;512]	384	27.3	8.8
D	256	256	27.0	8.8
	384	384	26.8	8.7
	[256;512]	384	25.6	8.1
E	256	256	27.3	9.0
	384	384	26.9	8.7
	[256;512]	384	**25.5**	**8.0**

LRN looks useless

1*1 fliter looks useless

Random S is better

Deeper is better

4. Classification experiment (2/4)

4.2 Multi-scale evaluation

Test scale $Q = \{S - 32, S, S + 32\}$ for models trained with fixed S, and $Q = \{S, 0.5(S_{min} + S_{max}), S_{max}\}$ for models trained with variable $S \in [S_{min}, S_{max}]$

Table 4: **ConvNet performance at multiple test scales.**

ConvNet config. (Table 1)	smallest image side		top-1 val. error (%)	top-5 val. error (%)
	train (S)	test (Q)		
B	256	224,256,288	28.2	9.6
C	256	224,256,288	27.7	9.2
	384	352,384,416	27.8	9.2
	[256; 512]	256,384,512	26.3	8.2
D	256	224,256,288	26.6	8.6
	384	352,384,416	26.5	8.6
	[256; 512]	256,384,512	**24.8**	**7.5**
E	256	224,256,288	26.9	8.7
	384	352,384,416	26.7	8.6
	[256; 512]	256,384,512	**24.8**	**7.5**

D, and E are best

Deeper is better

- scale jittering is better: indicate that scale jittering at test time leads to better performance (as compared to evaluating the same model at a single scale shown in Table 3).

4. Classification experiment (3/4)

Dense: Fully convolution + uncropped

4.3 Multi-crop evaluation

Dense and multi-crop are indeed complementary as their combination outperforms each of them

Table 5: **ConvNet evaluation techniques comparison.** In all experiments the training scale S was sampled from $[256; 512]$, and three test scales Q were considered: $\{256, 384, 512\}$.

ConvNet config. (Table 1)	Evaluation method	top-1 val. error (%)	top-5 val. error (%)
D	dense	24.8	7.5
	multi-crop	24.6	7.5
	multi-crop & dense	**24.4**	**7.2**
E	dense	24.8	7.5
	multi-crop	24.6	7.4
	multi-crop & dense	**24.4**	**7.1**

4.4 ConvNet Fusion

Dense and multi-crop are indeed complementary as their combination outperforms each of them

Table 6: **Multiple ConvNet fusion results.**

Combined ConvNet models	Error		
	top-1 val	top-5 val	top-5 test
S/Q ILSVRC submission			
(D/256/224,256,288), (D/384/352,384,416), (D/[256;512]/256,384,512) (C/256/224,256,288), (C/384/352,384,416) (E/256/224,256,288), (E/384/352,384,416)	24.7	7.5	7.3
post-submission			
(D/[256;512]/256,384,512), (E/[256;512]/256,384,512), dense eval.	24.0	7.1	7.0
(D/[256;512]/256,384,512), (E/[256;512]/256,384,512), multi-crop	23.9	7.2	-
(D/[256;512]/256,384,512), (E/[256;512]/256,384,512), multi-crop & dense eval.	**23.7**	**6.8**	**6.8**

Best result: we considered an ensemble of only two best-performing multi-scale models (configurations D and E) which reduced the test error to 7.0% using dense evaluation and 6.8%

4. Classification experiment (4/4)

4.5 Comparison with the state of the art

Table 7: **Comparison with the state of the art in ILSVRC classification**. Our method is denoted as "VGG". Only the results obtained without outside training data are reported.

Method	top-1 val. error (%)	top-5 val. error (%)	top-5 test error (%)
VGG (2 nets, multi-crop & dense eval.)	**23.7**	**6.8**	**6.8**
VGG (1 net, multi-crop & dense eval.)	24.4	7.1	7.0
VGG (ILSVRC submission, 7 nets, dense eval.)	24.7	7.5	7.3
GoogLeNet (Szegedy et al., 2014) (1 net)	-	7.9	
GoogLeNet (Szegedy et al., 2014) (7 nets)	-	**6.7**	
MSRA (He et al., 2014) (11 nets)	-	-	8.1
MSRA (He et al., 2014) (1 net)	27.9	9.1	9.1
Clarifai (Russakovsky et al., 2014) (multiple nets)	-	-	11.7
Clarifai (Russakovsky et al., 2014) (1 net)	-	-	12.5
Zeiler & Fergus (Zeiler & Fergus, 2013) (6 nets)	36.0	14.7	14.8
Zeiler & Fergus (Zeiler & Fergus, 2013) (1 net)	37.5	16.0	16.1
OverFeat (Sermanet et al., 2014) (7 nets)	34.0	13.2	13.6
OverFeat (Sermanet et al., 2014) (1 net)	35.7	14.2	-
Krizhevsky et al. (Krizhevsky et al., 2012) (5 nets)	38.1	16.4	16.4
Krizhevsky et al. (Krizhevsky et al., 2012) (1 net)	40.7	18.2	-

- Our best result is achieved by combining just two Networks – significantly less than used in most ILSVRC submissions.
- In terms of the single-net performance our architecture achieves the best result (7.0% test error) outperforming a single GoogLeNet by 0.9%.

5. Conclusions

In this work we evaluated very deep convolutional networks (up to 19 weight layers) for largescale image classification. It was demonstrated that the representation depth is beneficial for the classification accuracy.