Very Deep Convolutional Networks for Large-Scale Image Recognition

Karen Simonyan, Andrew Zisserman

Some slides are adopted from various sources.

Contents

- Convolutional Neural Networks
- Contributions
- Very Deep Convolutional Neural Networks
- Experiments & Results

Convolutional Neural Networks

A bit of Deep Learning history

- Perceptron algorithm developed by Rosenblatt in 1957.
- In 1970, Perceptron cannot approximate many nonlinear function (e.g., XOR)
- In 1980, multilayer perceptron is found to solve nonlinear decision boundary.
- In early 90’s, Back Propagation (BP) appeared.
- In 2006, a new wave of research on neural networks, and the field renamed to Deep Learning.

Why Deep Learning hot now?

Three driving factors ...

Big Data Availability

350 millions images uploaded per day

2.5 Petabytes of customer data hourly

100 hours of video uploaded every minute

New DL techniques

GPU Acceleration

Appearance of Large Scale Datasets

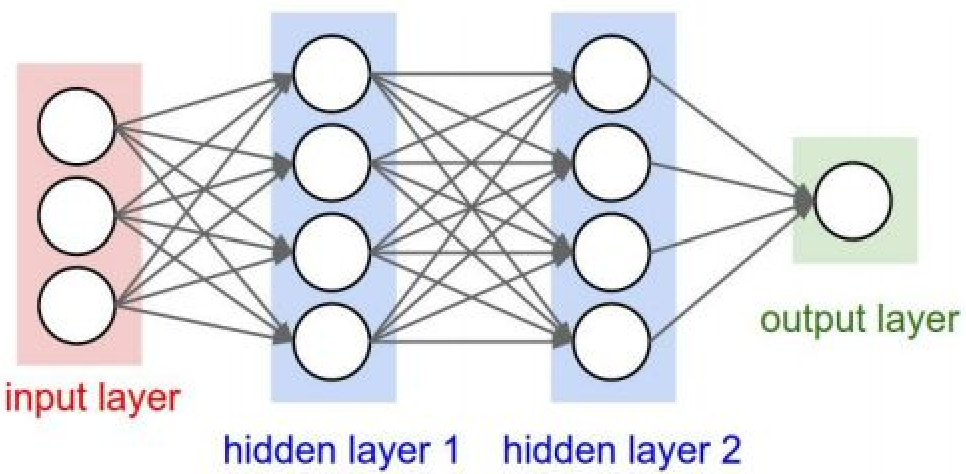
- ImageNet dataset
- ImageNet Large Scale Visual Recognition Challenge (ILSVRC)
- Places, the scene recognition database.
- Microsoft COCO, image recognition, segmentation, and captioning dataset.
- THUMOS, action recognition.

A machine learning algorithm usually corresponds to a combinations of the following 3 elements:

(either explicitly specified or implicit)

- ➢ The choice of a specific function family: F (often a parameterized family).
- ➢ A way to evaluate the quality of a function $f \in F$ (typically using a cost (or loss) function L measuring how wrongly *f* predicts).
- ➢ A way to search for the «best» function $f \in F$ (typically an optimization of function parameters to minimize the overall loss over the training set).

Function family F: Multilayer Neural Networks



Loss function L

- For classification problem, usually cross-entropy error function

$$J(\theta) = -\left[\sum_{i=1}^{m}\sum_{k=1}^{K} 1\left\{y^{(i)} = k\right\} \log P(y^{(i)} = k | x^{(i)}; \theta)\right]$$

Optimization: SGD + Backprobagation

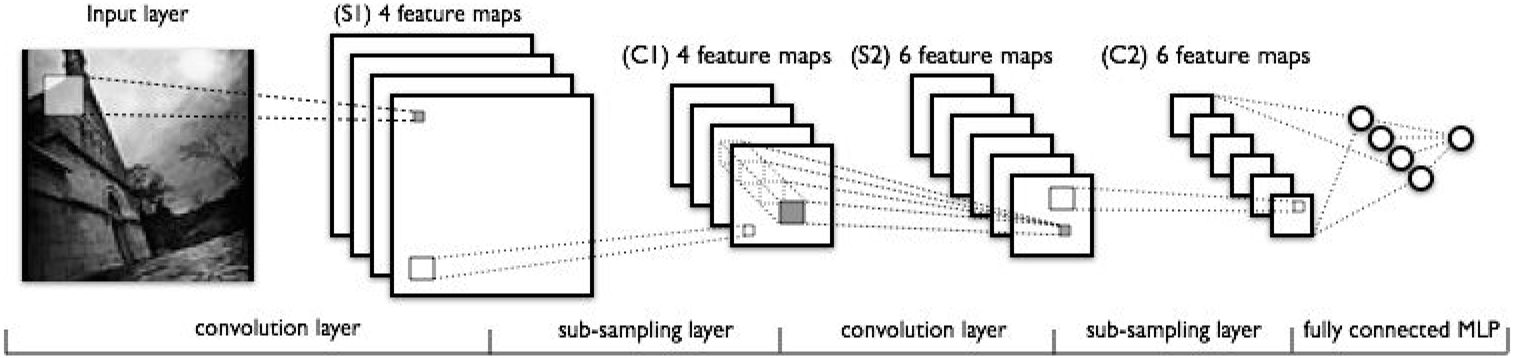
- SGD:
    - Mini-batch stochastic gradient descent
- Backprobagation:
    - An application of chain-rule

Convolutional Neural Networks

- Have at least 1 convolutional layer.
- Flagship network in Deep Learning
- Having been applied in some industry products.
    - Image search
    - Image retrieval
    - Image tagging, captioning.

Basic CNN architecture

- LeNet (LeCun *et. al.*)

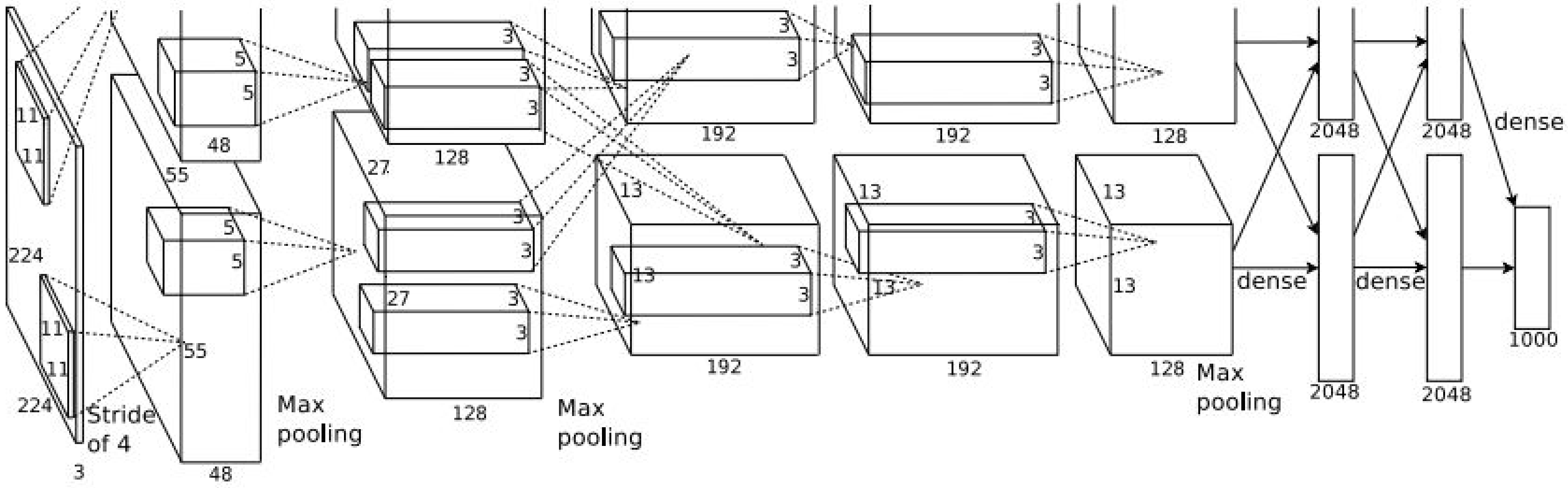


Modern CNNs

- Rectified Linear Units (ReLUs)
- Dropout
- Batch Normalization (BN)

Modern CNNs

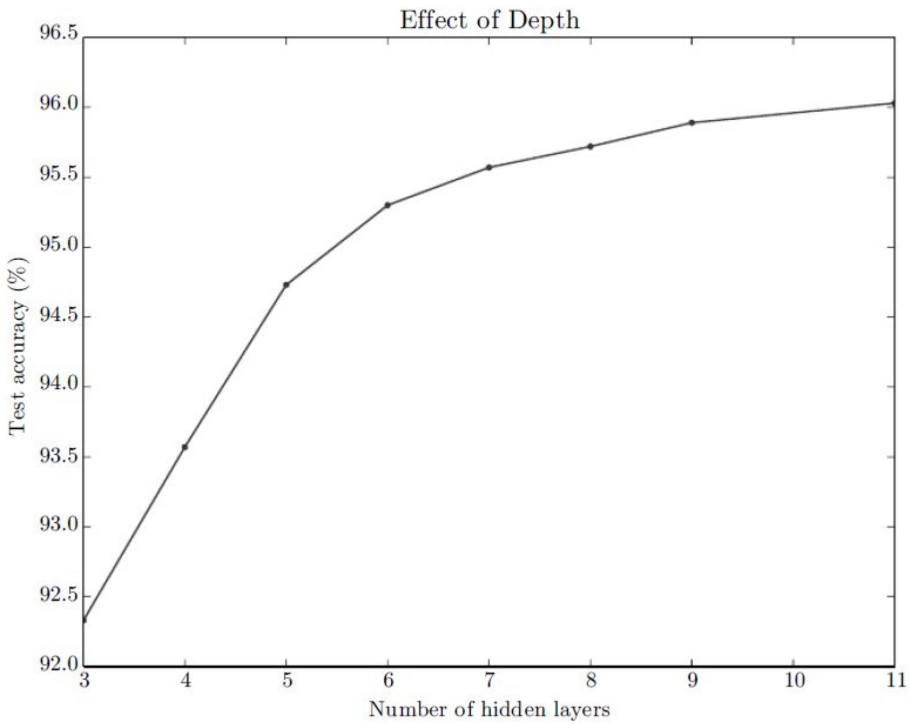
- AlexNet (Krizhevsky *et. al.*)



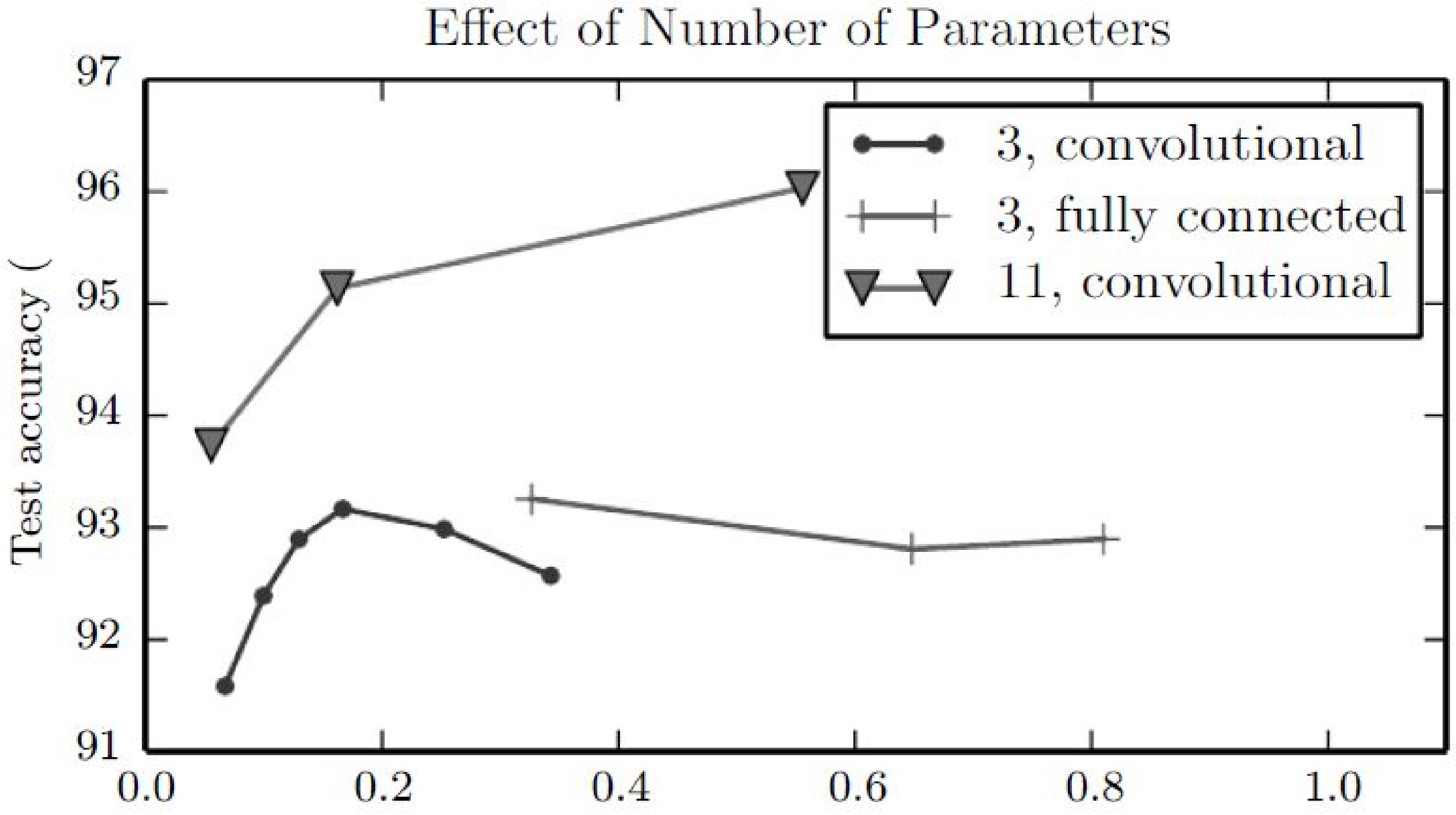
Universal approximation theorem

- Multilayer perceptrons are universal approximators.
- The universal approximation theorem means that regardless of what function we are trying to learn, we know that a large MLP will be able to represent this function.
- There are families of functions that can be represented efficiently by an architecture of depth k, but would require an exponential number of hidden units (with respect to the input size) with insufficient depth (depth 2 or depth $k-1$).

Effects of depth



Empirical results showing that deeper networks generalize better when used to transcribe multi-digit numbers from photographs of addresses ([Goodfellow et. al., 2014d](#))



Deeper models tend to perform better. This is not merely because the model is larger.
(Goodfellow et. al., 2014d)

Contributions

Contributions of the paper

- Empirically prove that deeper is better.

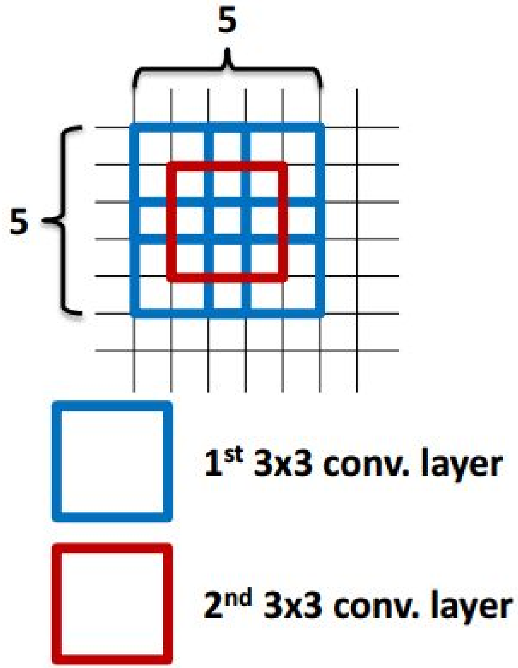
Very Deep Convolutional Neural Networks

ConvNet configuration

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					

3x3 filters

- Minimum size required for learning concepts of horizontal, vertical, blob.
- Stacking two 3x3 layers has an effective field of 5x5, stacking three 3x3 conv layers has an effective field of 7x7.
- Three 3x3 layers would have
    - Incorporating more nonlinear rectified layers, more discriminative
    - Implicit regularisation using deeper network
    - Less required parameters
    - 1 layer of 3x3 conv layers with C channels ⇒ 3*3*C = 9C
    - 3 layer of 3x3 conv layers with C channels ⇒ 27 C
    - 1 layer of 7x7 conv layers with C channels ⇒ 49 C



1x1 filters

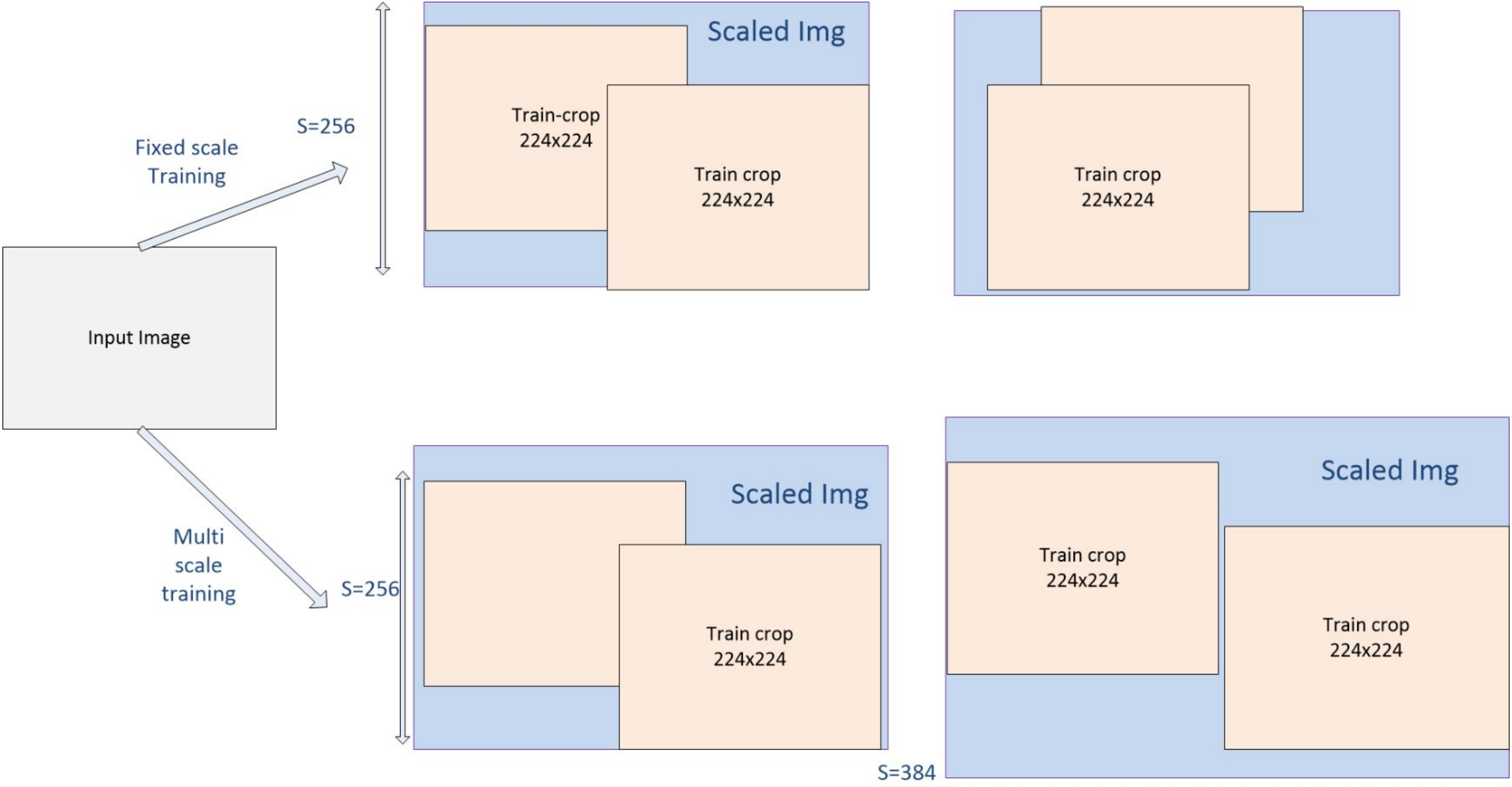
- Increase non-linearity w/o affecting receptive field
- Projection onto space of same dimension
- Used in Network in Network, GoogLeNet.

Training

- Minimising the multinomial logistic regression objective using mini-batch gradient descent
- Batch size 256
- Momentum 0.9
- Weight decay 0.0005
- Learning rate 0.01, decreasing by a factor of 10 when val accuracy stopped improving
- Learning stops after 370K iterations (74 epochs)
- Weights initialization: Shallow network with random initialisation, deeper networks from shallow network and random initialisation for deeper layers

Training

- Data augmentation:
  - Randomly cropped fixed-size 224x224
  - Random horizontal flip and random color shift
- Multi-scale training size
  - Two fixed scales: S = 256, S = 384
  - Variable scales: randomly sampling S from range [256, 512]
- Finetuning all layers of multi-scale model from single scale S=256 model.

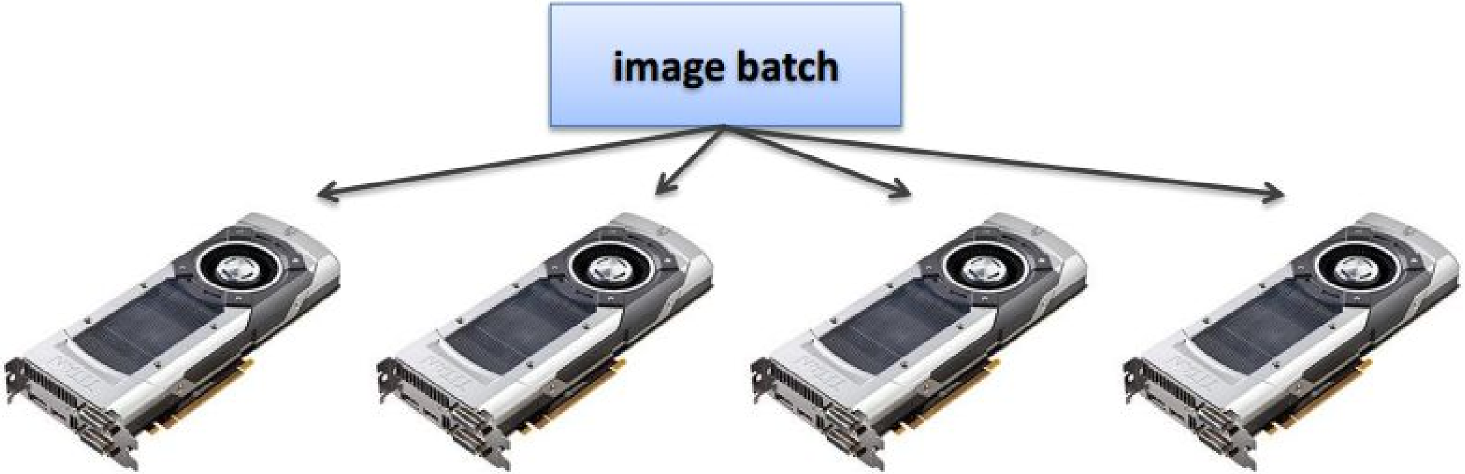

Input Image
Fixed scale
Training
S=256
Scaled Img
Train-crop
224x224
Train crop
224x224
Train crop
224x224
Multi
scale
training
S=256
Scaled Img
Train crop
224x224
S=384
Scaled Img
Train crop
224x224
Train crop
224x224

Testing

- Testing scale Q
- Each test image is scaled s/t smallest size Q
- Single scale evaluation: Q=S
- Multi scale evaluation: Try different Qs for a single S
- Multi-crop evaluation
  - 50 crops per scale
- Dense evaluation
  - FC layers converting to convolutional layers
  - FC-1000: 4096x1000 params into 1000 filters size 1x1x4096
- Average pooling to get final score

Implementation

- Modify BLVC Caffe
- Multi-GPU training
  - 4 NIVDIA Titan GPUs
  - Data parallelism for training and testing
  - 3.75 times speed-up, 2-3 weeks for training



Experiments & Results

Experiments: Single Test Scale Evaluation

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	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**

Experiments: Multiple test scales

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**

Observations

- Multi-scale at testing improves performance
- Multi-scale at training improves performance

Multi-crop vs dense evaluation

- Dense evaluation avoids recomputation for each crop
- Multi-crop slightly better
- Both are complementary perhaps due to different conv boundary conditions

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**

Ensembles

- Ensemble of 7 networks has error 7.3% error
- Ensemble of two-best performing multiscale models reduce test error to 7%
- Best performing single model 7.1% error
- Best results with ensemble of only 2 models 6.8% error

Final observations

- LRN does not improve performance
- Classification error decreases with increases ConvNet depth
- Important to capture more spatial context(config D vs C)
- Deepnets with small filters outperform shallow networks with large filters
    - Shallow version of B: 2 layers of 3x3 replaced with single 5x5 performs worse
- Error rate saturated at 19 layers
- Scale jittering at training helps capturing multiscale statistics and leads to better performance

VGG Net in ILSVRC

- First place in localization(25.3% error), second in classification(7.3% error) in ILSVRC 2014 using ensemble of 7 networks
- Outperforms Szegedy et.al(GoogLeNet) in terms of single network classification accuracy(7.1% vs 7.9%)

Deep Learning frameworks

- BLVC Caffe
  - Vision problems
  - Active community and developers
- Theano
  - Symbolic maths, more than just Deep Learning
  - Auto-differentiation
  - Wrappers: Keras, Lasage
- Torch
  - Twitter Cortex autograd (auto-diff)
- TensorFlow
  - Auto-differentiation
  - Faster time to production scale, native to Google Cloud Platform
- NVIDIA DIGITS:
  - A DL web app built on top of Caffe and Torch.

References

- Deep Learning book: http://www.deeplearningbook.org/
- Deep Learning summer school: http://videolectures.net/deeplearning2015_montreal/
- Deep learning tutorials: http://ufldl.stanford.edu/tutorial/
- Stanford deep learning for visual recognition course: http://cs231n.stanford.edu/