Return of the Devil in the Details: Delving Deep into Convolutional Nets

Ken Chatfield, Karen Simonyan, Andrea Vedaldi, and Andrew Zisserman
Visual Geometry Group, Department of Engineering Science, University of Oxford

Hilal E. Akyüz
The Devil is still in the Details

2011

The devil is in the details: an evaluation of recent feature encoding methods

Ken Chatfield
http://wwwrobots.ox.ac.uk/~ken

Victor Lampe
http://wwwrobots.ox.ac.uk/~vl

Andrea Vedaldi
http://wwwvfeat.org/~vedaldi

Andrew Zisserman
http://wwwrobots.ox.ac.uk/~az

Department of Engineering Science, Oxford University

Abstract

A large number of novel encodings for bag of visual words models have been proposed in the past two years to improve on the standard histogram of quantized local features. Examples include locality-constrained linear encoding [23], improved Fisher encoding [17], super vector encoding [27], and kernel codebook encoding [18]. While several authors have reported very good results on the challenging PASCAL VOC classification data by means of these new techniques, differences in the feature computation and learning algorithms, missing details in the description of the methods, and different tuning of the various components, make it impossible to compare directly these methods and hard to reproduce the results reported. This paper addresses these shortcomings by carrying out a rigorous evaluation of these new techniques: (1) fixing the other elements of the pipeline (features, learning, tuning), (2) disclosing all the implementation details, and (3) identifying both those aspects of each method which are particularly important to achieve good performance, and those aspects which are less critical. This allows a consistent comparative analysis of these encoding methods. Several conclusions drawn from our analysis cannot be inferred from the original publications.

1 Introduction

The typical object recognition pipeline is composed of the following three steps: (i) extraction of local image features (e.g., SIFT descriptors), (ii) encoding of the local features in an image descriptor (e.g., a histogram of the quantized local features), and (iii) classification of the bag of visual words.

2014

Return of the Devil in the Details: Delving Deep into Convolutional Nets

Ken Chatfield
ken@robots.ox.ac.uk

Karen Simonyan
karen@robots.ox.ac.uk

Andrea Vedaldi
vedaldi@robots.ox.ac.uk

Andrew Zisserman
az@robots.ox.ac.uk

Visual Geometry Group
Department of Engineering Science
University of Oxford
Oxford, UK

Abstract

The latest generation of Convolutional Neural Networks (CNN) have achieved impressive results in challenging benchmarks on image recognition and object detection, significantly raising the interest of the community in these methods. Nevertheless, it is still unclear how different CNN methods compare with each other and with previous state-of-the-art shallow representations such as the Bag-of-Visual-Words and the Improved Fisher Vectors. This paper contains a rigorous evaluation of these new techniques, exploring different deep architectures and comparing them on a common ground, identifying and discussing important implementation details. We identify several useful properties of CNN-based representations, including the fact that the dimensionality of the CNN output layer can be reduced significantly without having an adverse effect on performance. We also identify aspects of deep and shallow methods that can be successfully shared. In particular, we show that the data augmentation techniques commonly applied to CNN-based methods can also be applied to shallow methods, and result in an analogous performance boost. Source code and models to reproduce the experiments in the paper are made publicly available.

1 Introduction

Perhaps the single most important design choice in current state-of-the-art image classification and object recognition systems is the choice of visual features, or image representation. In fact, most of the quantitative improvements in image understanding obtained in the past decade years can be ascribed to the introduction of improved representations, from the Bag-of-Visual-Words (BoVW) [5, 28] to the Improved Fisher Vector (IFV) [23]. A common characteristic of these methods is that they are largely handcrafted. They are also relatively
Comparing Apples to Apples: State-of-the-art back in 2011

In our previous work (BMVC 2011) we conducted an extensive **evaluation of these encodings** comparing them all on a **common-ground**:

* we’ll call the features from these encodings **shallow** to distinguish them from the CNN-based features which follow
What is Changed Since 2011?

- Different deep architectures
- The latest generation of CNNs have achieved impressive results
- Unclear how the different methods introduced recently compare to each other and to shallow methods
Overview of the Paper

- This paper compares the latest (till 2014) methods on a common ground.
- Several properties of CNN-based representation and data augmentation techniques.
- Compare both different pre-trained network architectures and different learning heuristics.
Dataset (pre-training)

- **ILSVRC-2012**
  - Contains 1,000 object categories from ImageNet
  - ~1.2M training images
  - 50,000 validation images
  - 100,000 test images

- Performance is evaluated using **top-5** classification error
Datasets (training, fine-tuning)

- Pascal VOC 2007
  - Multi-label dataset
  - Contains ~10,000 images
  - 20 objects classes
  - Images split into train, validation and test sets.

- Pascal VOC 2012
  - Multi-label dataset
  - Contains ~ twice as many images
  - Does not include test set, instead, evaluation uses the official PASCAL Evaluation Server.

- Performance is measured as mean Average Precision (mAP)
Datasets (training, fine-tuning)

- **Caltech-101**
  - 101 classes
  - Three random split
  - 30 training, 30 testing images per class.

- **Caltech-256**
  - 256 classes
  - Two random split
  - 60 training, the rest are used for testing

- Performance is measured using mean class accuracy
Outline

- 3 scenarios:
  - Shallow representation
  - Deep representation (CNN) with pre-training
  - Deep representation (CNN) with pre-training and fine-tuning

- Different pre-trained networks
  - CNN-S, CNN-M, CNN-F

- Reducing CNN final layer output dimensionality
- Data augmentation (for both CNN and IFV)
- Color information
- Feature normalisation (for both CNN and IFV)
Data Augmentation

- Extract crops
- Pool features (average, max)

Pre-trained Network

CNN Feature Extractor
Data Augmentation

a. No augmentation (= 1 image)

b. Flip augmentation (= 2 images)

c. Crop+Flip augmentation (= 10 images)
Scenario 1: Shallow Representation (IFV)

- IFV usually **outperformed related encoding methods**

- Power normalization for improved
IFV Details

- Multi-scale dense sampling
- SIFT features
- Soft quantized using GMM with K=256 components
- Spatial Pyramid (1x1, 3x1, 2x2)
- 3 modification:
  - Intra-norm
    - L2 norm is applied to the subblocks
  - Spatially-extended local descriptors
    - Memory-efficient than SPM
  - Color features
    - Local Color Statistics
Scenario 2: Deep Representation (CNN) with Pre-training

- Pre-trained on ImageNet

- 3 different pre-trained networks
### Pre-trained Networks

- **CNN-F** similar to Krizhevsky et al., NIPS 2012: `ImageNet classification with deep convolutional networks`
  - **conv1**: $64 \times 11 \times 11$, stride 4
  - **conv2**: $256 \times 5 \times 5$, stride 1
  - **conv3**: $256 \times 3 \times 3$, stride 1
  - **conv4**: $256 \times 3 \times 3$
  - **conv5**: $256 \times 3 \times 3$
  - **fc6**: 4096 d.o.
  - **fc7**: 4096 drop-out

- **CNN-M** similar to Zeiler and Fergus, CoRR 2013: `Visualising and understanding convolutional networks`
  - **conv1**: $96 \times 7 \times 7$, stride 2
  - **conv2**: $256 \times 5 \times 5$, stride 2
  - **conv3**: $512 \times 3 \times 3$, stride 1
  - **conv4**: $512 \times 3 \times 3$
  - **conv5**: $512 \times 3 \times 3$
  - **fc6**: 4096 d.o.
  - **fc7**: 4096 drop-out

- **CNN-S** similar to OverFeat ‘accurate’ network, ICLR 2014: `OverFeat: integrated recognition, localisation and detection using ConvNets`
  - **conv1**: $96 \times 7 \times 7$, stride 2
  - **conv2**: $256 \times 5 \times 5$, stride 1
  - **conv3**: $512 \times 3 \times 3$, stride 1
  - **conv4**: $512 \times 3 \times 3$
  - **conv5**: $512 \times 3 \times 3$
  - **fc6**: 4096 d.o.
  - **fc7**: 4096 drop-out
# Pre-Trained Networks

<table>
<thead>
<tr>
<th>Arch.</th>
<th>conv1</th>
<th>conv2</th>
<th>conv3</th>
<th>conv4</th>
<th>conv5</th>
<th>full6</th>
<th>full7</th>
<th>full8</th>
</tr>
</thead>
<tbody>
<tr>
<td>CNN-F</td>
<td>64x11x11</td>
<td>256x5x5</td>
<td>256x3x3</td>
<td>256x3x3</td>
<td>256x3x3</td>
<td>4096</td>
<td>4096</td>
<td>1000</td>
</tr>
<tr>
<td></td>
<td>st. 4, pad 0</td>
<td>st. 1, pad 2</td>
<td>st. 1, pad 1</td>
<td>st. 1, pad 1</td>
<td>st. 1, pad 1</td>
<td>dropout</td>
<td>dropout</td>
<td>softmax</td>
</tr>
<tr>
<td></td>
<td>LRN, x2 pool</td>
<td>LRN, x2 pool</td>
<td>-</td>
<td>-</td>
<td>x2 pool</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>CNN-M</td>
<td>96x7x7</td>
<td>256x5x5</td>
<td>512x3x3</td>
<td>512x3x3</td>
<td>512x3x3</td>
<td>4096</td>
<td>4096</td>
<td>1000</td>
</tr>
<tr>
<td></td>
<td>st. 2, pad 0</td>
<td>st. 2, pad 1</td>
<td>st. 1, pad 1</td>
<td>st. 1, pad 1</td>
<td>st. 1, pad 1</td>
<td>dropout</td>
<td>dropout</td>
<td>softmax</td>
</tr>
<tr>
<td></td>
<td>LRN, x2 pool</td>
<td>LRN, x2 pool</td>
<td>-</td>
<td>-</td>
<td>x2 pool</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>CNN-S</td>
<td>96x7x7</td>
<td>256x5x5</td>
<td>512x3x3</td>
<td>512x3x3</td>
<td>512x3x3</td>
<td>4096</td>
<td>4096</td>
<td>1000</td>
</tr>
<tr>
<td></td>
<td>st. 2, pad 0</td>
<td>st. 1, pad 1</td>
<td>st. 1, pad 1</td>
<td>st. 1, pad 1</td>
<td>st. 1, pad 1</td>
<td>dropout</td>
<td>dropout</td>
<td>softmax</td>
</tr>
<tr>
<td></td>
<td>LRN, x3 pool</td>
<td>x2 pool</td>
<td>-</td>
<td>-</td>
<td>x3 pool</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Scenario 3: Deep Representation (CNN) with Pre-training & Fine-tuning

- Pre-trained on one dataset and applied to another
- Improve the performance
- Become dataset-specific
CNN Details

- Trained with same training protocol, same implementation
- Caffe framework
- L2 normalization of CNN features
  - Before introducing to SVM
CNN Training

• Gradient descent with momentum
  - Momentum is 0.9
  - Weight decay is $5 \times 10^{-4}$
  - Learning rate is $10^{-2}$, decreased by 10

• Data augmentation
  - Random crops
  - Flips
  - RGB jitterring

• 3 weeks with a Titan Black (Slow arch.)
CNN Fine-tuning

- Only last layer

- Classification hinge loss (CNN-S TUNE-CLS), ranking hinge loss (CNN-S TUNE-RNK) for VOC

- Softmax regression loss for Caltech-101

- Lower initial learning rate (VOC & Caltech)
Low Dimensional CNN Features

- Baseline networks all have 4096-D last hidden layer
- We further trained three modifications to CNN-M with lower dimensional full7 layers

* Note: as only the original ILSVRC-2012 data was used for re-training this differs from fine-tuning and is simply a way of reducing the final output dimension
Analysis
Pre-trained Networks

mAP (VOC07)

- Decaf: 73.41
- CNN-F: 77.38
- CNN-M: 79.89
- CNN-S: 79.74

80
78.25
76.5
74.75
73
Data Augmentation

- None
- Flip
- Crop+Flip (train pooling: sum, test pooling: sum)
- Crop+Flip (train pooling: none, test pooling: sum)

mAP (VOC07)

IFV
- 64.36
- 64.35
- 66.68
- 67.17

CNN-M
- 76.97
- 76.99
- 79.44
- 79.89
Impact of Colour

- Greyscale
- Greyscale+aug
- Colour
- Colour+aug

mAP (VOC07)

IFV-512
- 65.36
- 66.37
- 68.02
- 67.93

CNN-M
- 73.59
- 76.97
- 77
- 79.89
Low Dimensional CNN Features

mAP (VOC07)

CNN-M

4096 2048 1024 128

79.89 80.1 79.91 78.6
• Our best CNN method achieves state-of-the-art performance over several datasets

• How do we get there? through comparison on equal footing, we determine what’s important and what’s not

slide by Chatfield et al
<table>
<thead>
<tr>
<th>Method</th>
<th>SPool</th>
<th>Image</th>
<th>Aug.</th>
<th>Dim</th>
<th>mAP</th>
</tr>
</thead>
<tbody>
<tr>
<td>(I) FK BL</td>
<td>spm</td>
<td>–</td>
<td></td>
<td>327K</td>
<td>61.69</td>
</tr>
<tr>
<td>(II) DECAFE</td>
<td>–</td>
<td>(C)</td>
<td>t t</td>
<td>327K</td>
<td>73.41</td>
</tr>
<tr>
<td>(a) FK</td>
<td>spm</td>
<td>–</td>
<td></td>
<td>327K</td>
<td>63.66</td>
</tr>
<tr>
<td>(b) FK IN</td>
<td>spm</td>
<td>–</td>
<td></td>
<td>327K</td>
<td>64.18</td>
</tr>
<tr>
<td>(c) FK</td>
<td>(x,y)</td>
<td>–</td>
<td></td>
<td>42K</td>
<td>63.51</td>
</tr>
<tr>
<td>(d) FK IN</td>
<td>(x,y)</td>
<td>–</td>
<td></td>
<td>42K</td>
<td>64.36</td>
</tr>
<tr>
<td>(e) FK IN</td>
<td>(x,y)</td>
<td>(F)</td>
<td>f</td>
<td>42K</td>
<td>64.35</td>
</tr>
<tr>
<td>(f) FK IN</td>
<td>(x,y)</td>
<td>(C)</td>
<td>f s</td>
<td>42K</td>
<td>67.17</td>
</tr>
<tr>
<td>(g) FK IN</td>
<td>(x,y)</td>
<td>(C)</td>
<td>s s</td>
<td>42K</td>
<td>66.68</td>
</tr>
<tr>
<td>(h) FK IN 512</td>
<td>(x,y)</td>
<td>–</td>
<td></td>
<td>84K</td>
<td>65.36</td>
</tr>
<tr>
<td>(i) FK IN 512</td>
<td>(x,y)</td>
<td>(C)</td>
<td>f s</td>
<td>84K</td>
<td>68.02</td>
</tr>
<tr>
<td>(j) FK IN COL 512</td>
<td>(x,y)</td>
<td>–</td>
<td></td>
<td>82K</td>
<td>52.18</td>
</tr>
<tr>
<td>(k) FK IN 512 COL+</td>
<td>(x,y)</td>
<td>–</td>
<td></td>
<td>166K</td>
<td>66.37</td>
</tr>
<tr>
<td>(l) FK IN 512 COL+</td>
<td>(x,y)</td>
<td>(C)</td>
<td>f s</td>
<td>166K</td>
<td>67.93</td>
</tr>
<tr>
<td>(m) CNN F</td>
<td>–</td>
<td>(C)</td>
<td>f s</td>
<td>4K</td>
<td>77.38</td>
</tr>
<tr>
<td>(n) CNN S</td>
<td>–</td>
<td>(C)</td>
<td>f s</td>
<td>4K</td>
<td>79.74</td>
</tr>
<tr>
<td>(o) CNN M</td>
<td>–</td>
<td>–</td>
<td></td>
<td>4K</td>
<td>76.97</td>
</tr>
<tr>
<td>(p) CNN M</td>
<td>–</td>
<td>(C)</td>
<td>f s</td>
<td>4K</td>
<td>79.89</td>
</tr>
<tr>
<td>(q) CNN M</td>
<td>–</td>
<td>(C)</td>
<td>f m</td>
<td>4K</td>
<td>79.50</td>
</tr>
<tr>
<td>(r) CNN M</td>
<td>–</td>
<td>(C)</td>
<td>s s</td>
<td>4K</td>
<td>79.44</td>
</tr>
<tr>
<td>(s) CNN M</td>
<td>–</td>
<td>(C)</td>
<td>t t</td>
<td>41K</td>
<td>78.77</td>
</tr>
<tr>
<td>(t) CNN M</td>
<td>–</td>
<td>(C)</td>
<td>f</td>
<td>4K</td>
<td>77.78</td>
</tr>
<tr>
<td>(u) CNN M</td>
<td>–</td>
<td>(F)</td>
<td>f</td>
<td>4K</td>
<td>76.99</td>
</tr>
<tr>
<td>(v) CNN M GS</td>
<td>–</td>
<td>–</td>
<td></td>
<td>4K</td>
<td>73.59</td>
</tr>
<tr>
<td>(w) CNN M GS</td>
<td>–</td>
<td>(C)</td>
<td>f s</td>
<td>4K</td>
<td>77.00</td>
</tr>
<tr>
<td>(x) CNN M 2048</td>
<td>–</td>
<td>(C)</td>
<td>f s</td>
<td>2K</td>
<td>80.10</td>
</tr>
<tr>
<td>(y) CNN M 1024</td>
<td>–</td>
<td>(C)</td>
<td>f s</td>
<td>1K</td>
<td>79.91</td>
</tr>
<tr>
<td>(z) CNN M 128</td>
<td>–</td>
<td>(C)</td>
<td>f s</td>
<td>128</td>
<td>78.60</td>
</tr>
<tr>
<td>(α) FK+CNN F</td>
<td>(x,y)</td>
<td>(C)</td>
<td>f s</td>
<td>88K</td>
<td>77.95</td>
</tr>
<tr>
<td>(β) FK+CNN M 2048</td>
<td>(x,y)</td>
<td>(C)</td>
<td>f s</td>
<td>86K</td>
<td>80.14</td>
</tr>
<tr>
<td>(γ) CNN S TUNE-RNK</td>
<td>–</td>
<td>(C)</td>
<td>f s</td>
<td>4K</td>
<td>82.42</td>
</tr>
<tr>
<td></td>
<td>ILSVRC-2012 (top-5 error)</td>
<td>VOC-2007 (mAP)</td>
<td>VOC-2012 (mAP)</td>
<td>Caltech-101 (accuracy)</td>
<td>Caltech-256 (accuracy)</td>
</tr>
<tr>
<td>----------------</td>
<td>---------------------------</td>
<td>----------------</td>
<td>----------------</td>
<td>------------------------</td>
<td>------------------------</td>
</tr>
<tr>
<td>(a) FK IN 512</td>
<td>-</td>
<td>68.0</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>(b) CNN F</td>
<td>16.7</td>
<td>77.4</td>
<td>79.9</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>(c) CNN M</td>
<td>13.7</td>
<td>79.9</td>
<td>82.5</td>
<td>87.15 ± 0.80</td>
<td>77.03 ± 0.46</td>
</tr>
<tr>
<td>(d) CNN M 2048</td>
<td>13.5</td>
<td>80.1</td>
<td>82.4</td>
<td>86.64 ± 0.53</td>
<td>76.88 ± 0.35</td>
</tr>
<tr>
<td>(e) CNN S</td>
<td><strong>13.1</strong></td>
<td>79.7</td>
<td>82.9</td>
<td>87.76 ± 0.66</td>
<td><strong>77.61 ± 0.12</strong></td>
</tr>
<tr>
<td>(f) CNN S TUNE-CLS</td>
<td><strong>13.1</strong></td>
<td>-</td>
<td>83.0</td>
<td><strong>88.35 ± 0.56</strong></td>
<td>77.33 ± 0.56</td>
</tr>
<tr>
<td>(g) CNN S TUNE-RNK</td>
<td><strong>13.1</strong></td>
<td><strong>82.4</strong></td>
<td><strong>83.2</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>(h) Zeiler &amp; Fergus [23]</td>
<td>16.1</td>
<td>-</td>
<td>79.0</td>
<td>86.5 ± 0.5</td>
<td>74.2 ± 0.3</td>
</tr>
<tr>
<td>(i) Razavian et al. [24, 25]</td>
<td>14.7</td>
<td>77.2</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>(j) Oquab et al. [26]</td>
<td>18</td>
<td>77.7</td>
<td>78.7 (82.8*)</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>(k) Oquab et al. [26]</td>
<td>-</td>
<td>-</td>
<td>86.3*</td>
<td></td>
<td></td>
</tr>
<tr>
<td>(l) Wei et al. [27]</td>
<td>-</td>
<td>81.5 (85.2*)</td>
<td>81.7 (90.3*)</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>(m) He et al. [28]</td>
<td>13.6</td>
<td>80.1</td>
<td>-</td>
<td>91.4 ± 0.7</td>
<td>-</td>
</tr>
</tbody>
</table>

Table 3: **Comparison with the state of the art** on ILSVRC2012, VOC2007, VOC2012, Caltech-101, and Caltech-256. Results marked with * were achieved using models pre-trained on the extended ILSVRC datasets (1512 classes in [24, 25], 2000 classes in [30]). All other results were achieved using CNNs pre-trained on ILSVRC-2012 (1000 classes).
## Comparison to State-of-the-art

<table>
<thead>
<tr>
<th>Model</th>
<th>ILSVRC-2012</th>
<th>VOC2007</th>
<th>VOC2012</th>
</tr>
</thead>
<tbody>
<tr>
<td>CNN-M 2048</td>
<td>13.5</td>
<td>80.1</td>
<td>82.4</td>
</tr>
<tr>
<td>CNN-S</td>
<td>13.1</td>
<td>79.7</td>
<td>82.9</td>
</tr>
<tr>
<td>CNN-S TUNE-RNK</td>
<td>13.1</td>
<td>82.4</td>
<td>83.2</td>
</tr>
<tr>
<td>Zeiler &amp; Fergus</td>
<td>16.1</td>
<td></td>
<td>79.0</td>
</tr>
<tr>
<td>Oquab et al.</td>
<td>18.0</td>
<td>77.7</td>
<td>78.7 (82.8*)</td>
</tr>
<tr>
<td>Oquab et al.</td>
<td></td>
<td></td>
<td>86.3*</td>
</tr>
<tr>
<td>Wei et al.</td>
<td></td>
<td>81.5 (85.2*)</td>
<td>81.7 (90.3*)</td>
</tr>
</tbody>
</table>

* Uses extended training data and/or fusion with other methods
Take Home Messages

• **Data augmentation** helps a lot, both for deep and shallow methods

• **Fine-tuning** makes a difference, and use of **ranking loss** can be preferred

• Smaller filters and deeper networks help, although feature computation is slower

• **CNN-based methods >> shallow methods**

• We can transfer tricks from deep features to shallow features

• We can achieve incredibly low dimensional (~128D) but performant features with CNN-based methods

• **If you get the details right, it's possible to get to state-of-the-art with very simple methods!!**
One more thing...

- CNN models and feature computation code can now be downloaded from the project website: http://www.robots.ox.ac.uk/~vgg/software/deep_eval/
- As before, source code to reproduce all experiments will be made available
Thank You For Listening..

Q&A?
(DEMO)

Hilal E. Akyüz
<table>
<thead>
<tr>
<th>CNN Model</th>
<th>Pascal VOC 2007 mAP</th>
</tr>
</thead>
<tbody>
<tr>
<td>CNN-S</td>
<td>76.10</td>
</tr>
<tr>
<td>CNN-M</td>
<td>76.11</td>
</tr>
<tr>
<td>AlexNet</td>
<td>71.40</td>
</tr>
<tr>
<td>GoogleNet</td>
<td>80.91</td>
</tr>
<tr>
<td>ResNet</td>
<td><strong>83.06</strong></td>
</tr>
<tr>
<td>VGG19</td>
<td>81.01</td>
</tr>
</tbody>
</table>
## Demo

<table>
<thead>
<tr>
<th>Model</th>
<th>FPS (batch size=1)</th>
</tr>
</thead>
<tbody>
<tr>
<td>CNN_M</td>
<td>169</td>
</tr>
<tr>
<td>CNN_S</td>
<td>151</td>
</tr>
<tr>
<td>ResNet</td>
<td>11</td>
</tr>
<tr>
<td>GoogleNet</td>
<td>71</td>
</tr>
<tr>
<td>VGG19</td>
<td>50</td>
</tr>
</tbody>
</table>
Extras

VGG Very Deep Network
Simonyan & Zisserman (ICLR 2015)

Smaller receptive window size + stride, and deeper
Extras

![Bar chart showing mAP (VOC07) results for different models: Decaf, CNN-F, CNN-M, CNN-S, and VGG-VD. The values are 73.41, 77.38, 79.89, 79.74, and 89.3 respectively.](image)
### Comparison with State of the Art

<table>
<thead>
<tr>
<th>Model</th>
<th>ILSVRC-2012</th>
<th>VOC2007</th>
<th>VOC2012</th>
</tr>
</thead>
<tbody>
<tr>
<td>CNN-M 2048</td>
<td>13.5</td>
<td>80.1</td>
<td>82.4</td>
</tr>
<tr>
<td>CNN-S</td>
<td>13.1</td>
<td>79.7</td>
<td>82.9</td>
</tr>
<tr>
<td>CNN-S TUNE-RNK</td>
<td>13.1</td>
<td>82.4</td>
<td>83.2</td>
</tr>
<tr>
<td>Zeiler &amp; Fergus</td>
<td>16.1</td>
<td></td>
<td>79.0</td>
</tr>
<tr>
<td>Oquab et al.</td>
<td>18.0</td>
<td>77.7</td>
<td>78.7 (82.8*)</td>
</tr>
<tr>
<td>Wei et al.</td>
<td></td>
<td>81.5 (85.2*)</td>
<td>81.7 (90.3*)</td>
</tr>
<tr>
<td>Clarifai (1 net)</td>
<td>12.5</td>
<td></td>
<td></td>
</tr>
<tr>
<td>GoogLeNet (1 net)</td>
<td>7.9</td>
<td></td>
<td></td>
</tr>
<tr>
<td>VGG Very Deep (1 net)</td>
<td>7.0</td>
<td>89.3</td>
<td>89.0</td>
</tr>
</tbody>
</table>