In-Depth Case Study: Residual Network

Joe Yeh, M.D.

Computer Science > Computer Vision and Pattern Recognition

Deep Residual Learning for Image Recognition

Kaiming He, Xiangyu Zhang, Shaoqing Ren, Jian Sun

(Submitted on 10 Dec 2015)

Deeper neural networks are more difficult to train. We present a residual learning framework to ease the training of networks that are substantially deeper than those used previously. We explicitly reformulate the layers as learning residual functions with reference to the layer inputs, instead of learning unreferenced functions. We provide comprehensive empirical evidence showing that these residual networks are easier to optimize, and can gain accuracy from considerably increased depth. On the ImageNet dataset we evaluate residual nets with a depth of up to 152 layers---8x deeper than VGG nets but still having lower complexity. An ensemble of these residual nets achieves 3.57% error on the ImageNet test set. This result won the 1st place on the ILSVRC 2015 classification task. We also present analysis on CIFAR-10 with 100 and 1000 layers.

The depth of representations is of central importance for many visual recognition tasks. Solely due to our extremely deep representations, we obtain a 28% relative improvement on the COCO object detection dataset. Deep residual nets are foundations of our submissions to ILSVRC & COCO 2015 competitions, where we also won the 1st places on the tasks of ImageNet detection, ImageNet localization, COCO detection, and COCO segmentation.

Comments: Tech report
Subjects: **Computer Vision and Pattern Recognition (cs.CV)**
Cite as: **arXiv:1512.03385 [cs.CV]**
(or **arXiv:1512.03385v1 [cs.CV]** for this version)

Background

- Deep CNNs have led to breakthroughs in image recognition.
- CNNs naturally integrate hierarchy of image features -> deeper nets capture more levels of details
- Recent evidence shows deeper nets achieve better results
- However…

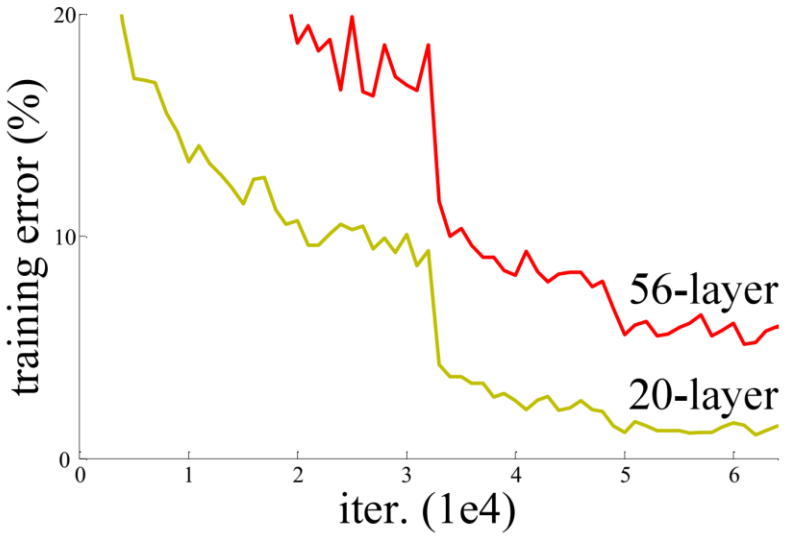
Problem

- While deep CNNs are able to start converging, degradation begins to show -> with depth increasing, accuracy saturates and then degrades rapidly
- Such degradation is not due to *overfitting*, and adding more layers may lead to higher training error.

Analysis of the Problem

- Let us consider a shallower architecture and its deeper counterpart.
- We know that by purposefully adding a few layers that are identity mapping to a well-trained shallower architecture, we end up with a deeper architecture that performs exactly the same.
- The existence of this constructed solution indicates that a deeper model should produce no higher training error than its shallower counterpart.
- But experiments show that our current solvers on hand are unable to find solutions that are comparably good or better than the constructed solution (or unable to do so in feasible time).

Experiment on Network Depth



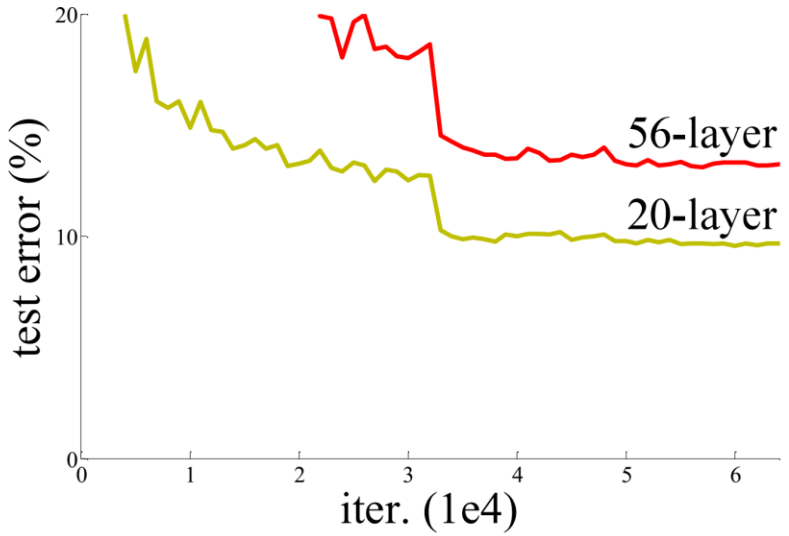


Figure 1. Training error (left) and test error (right) on CIFAR-10 with 20-layer and 56-layer "plain" networks. The deeper network has higher training error, and thus test error. Similar phenomena on ImageNet is presented in Fig. 4.

Proposed Solution

- Instead of hoping each few stacked layers directly fit a desired underlying mapping, we explicitly let these layers fit a residual mapping.
- Formally, denoting the desired underlying mapping as $H(\mathbf{x})$, we let the stacked nonlinear layers fit another mapping of $F(\mathbf{x}) := H(\mathbf{x}) - \mathbf{x}$.
- The original mapping is recast into $F(\mathbf{x})+\mathbf{x}$.
- We hypothesize that it is easier to optimize the residual mapping than to optimize the original, unreferenced mapping.
- To the extreme, if an identity mapping were optimal, it would be easier to push the residual to zero than to fit an identity mapping by a stack of nonlinear layers.

Proposed Solution

- The formulation of $F(\mathbf{x})+\mathbf{x}$ can be realized by feedforward neural networks with "shortcut connections"
- In our case, the shortcut connections simply perform *identity* mapping, and their outputs are added to the outputs of the stacked layers.
- Identity shortcut connections add neither extra parameter nor computational complexity.

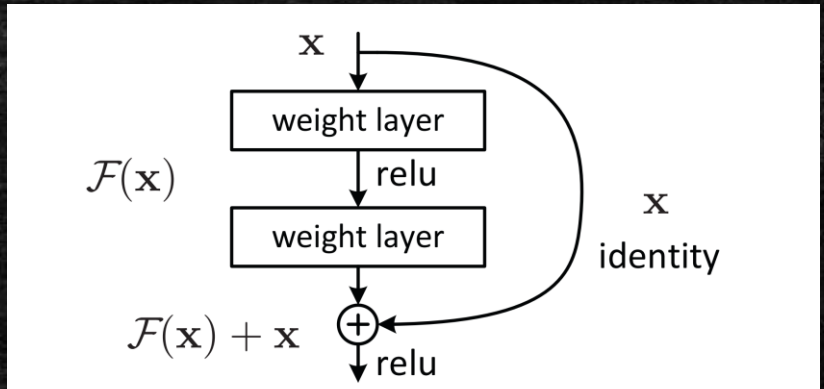


Figure 2. Residual learning: a building block.

Related work : Residual Representation

- Vector of Locally Aggregated Descriptors (VLAD):
- Regions are extracted from an image using an affine invariant detector, and described using 128-D SIFT descriptor. Each descriptor is then assigned to the closest cluster of a vocabulary of size k (typically 64 or 256, so that clusters are quite coarse). For each of the k clusters, the residuals (vector differences between descriptors and cluster centers) are accumulated, and the k 128-D sums of residuals are concatenated into a single k × 128 dimensional descriptor.
- Ref : H. Jegou, M. Douze, C. Schmid, and P. Perez, "Aggregating local descriptors into a compact image representation" in CVPR, June 2010. https://lear.inrialpes.fr/pubs/2010/JDSP10/jegou_compactimagerepresentation.pdf
- SIFT : Scale Invariant Feature Transform

Related work : Residual Representation

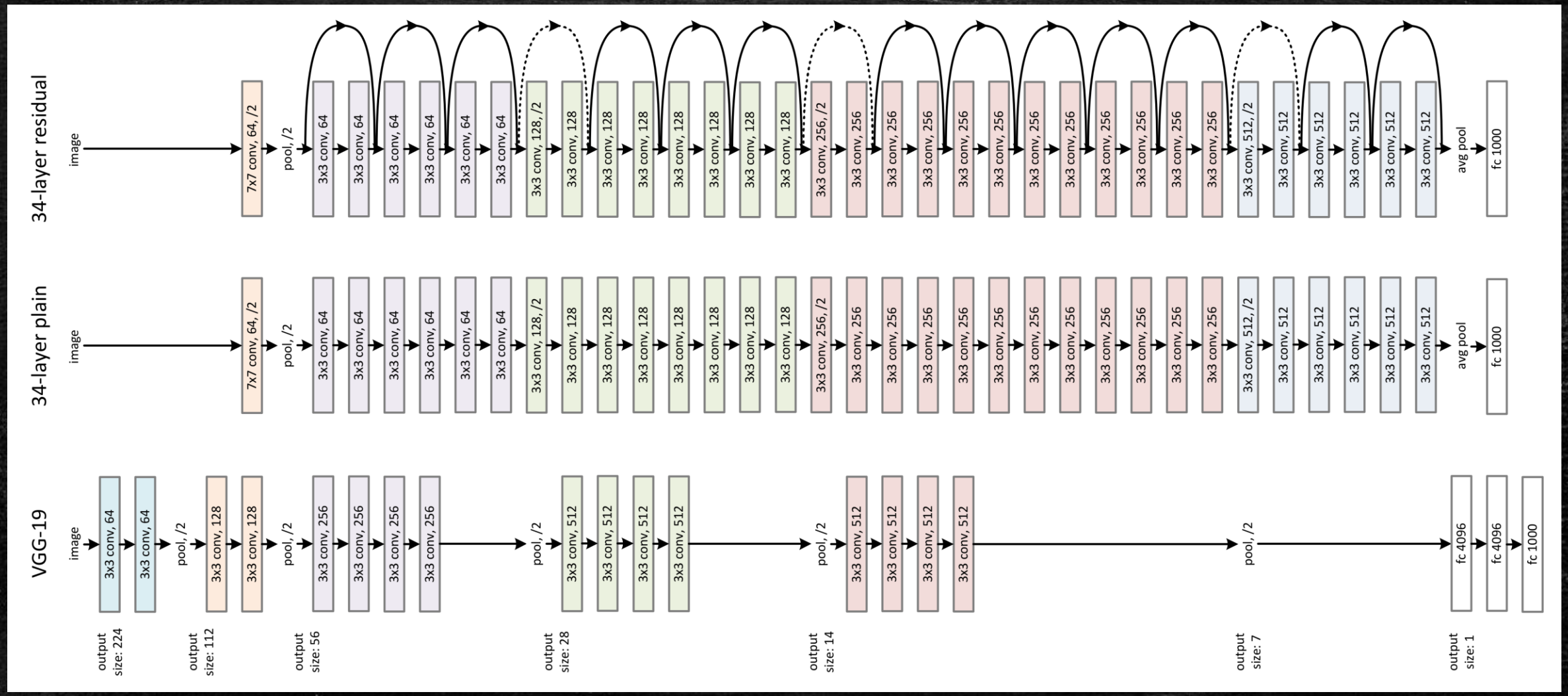
- Fisher Vector:
- A Gaussian Mixture Model (GMM) is used to model the distribution of features (e.g. SIFT) extracted all over the image
- The Fisher Vector (FV) encodes the gradients of the log-likelihood of the features under the GMM, with respect to the GMM parameters.

- Ref : F. Perronnin and C. Dance. Fisher kernels on visual vocabularies for image categorization. In *CVPR*, 2007 http://www.europe.naverlabs.com/layout/set/print/content/download/20837/148502/file/2006-034.pdf

SIFT : Scale-Invariant Feature Transform

- 1. Scale-space extrema detection: The first stage of computation searches over all scales and image locations. It is implemented efficiently by using a difference-of-Gaussian function to identify potential interest points that are invariant to scale and orientation.
- 2. Keypoint localization: At each candidate location, a detailed model is fit to determine location and scale. Keypoints are selected based on measures of their stability.
- 3. Orientation assignment: One or more orientations are assigned to each keypoint location based on local image gradient directions. All future operations are performed on image data that has been transformed relative to the assigned orientation, scale, and location for each feature, thereby providing invariance to these transformations.
- 4. Keypoint descriptor: The local image gradients are measured at the selected scale in the region around each keypoint. These are transformed into a representation that allows for significant levels of local shape distortion and change in illumination.
- Ref: Lowe, 2004, Distinctive Image Features from Scale-Invariant Keypoints https://people.eecs.berkeley.edu/~malik/cs294/lowe-ijcv04.pdf

Network Architecture of ResNet



ImageNet Training Result

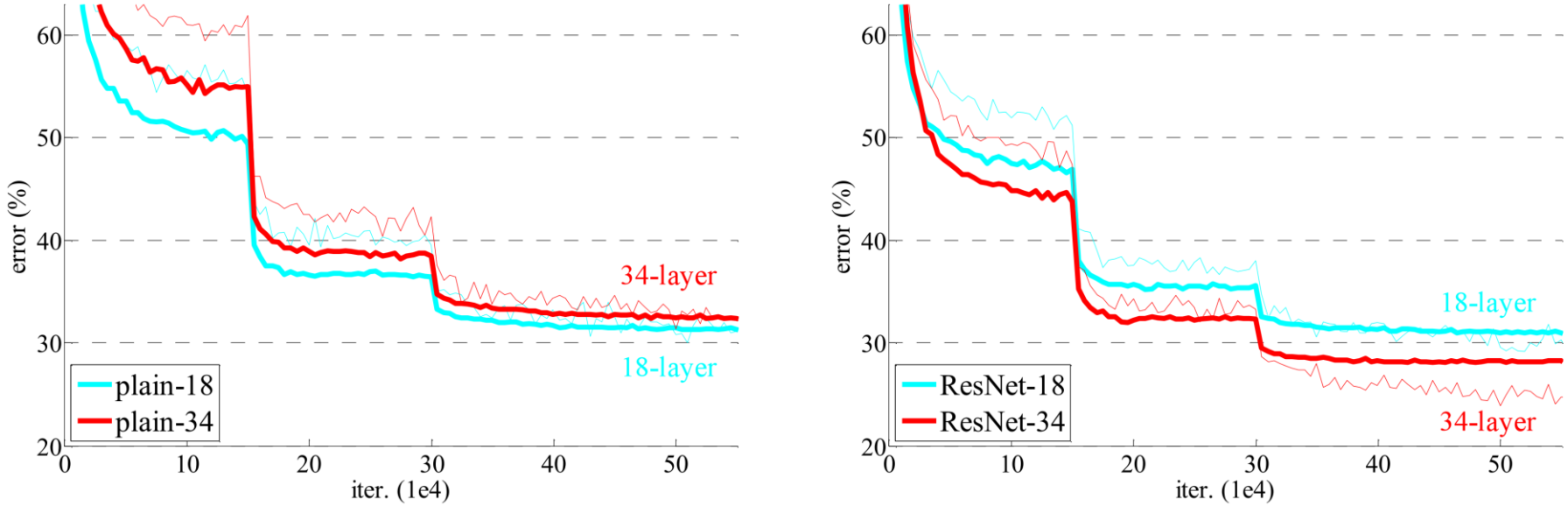


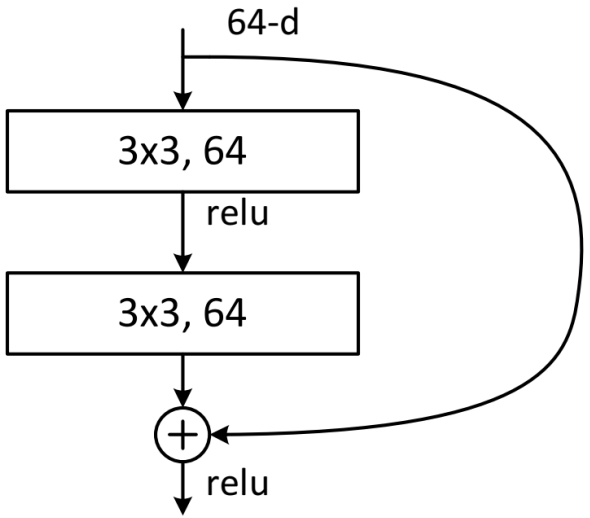
Figure 4. Training on **ImageNet**. Thin curves denote training error, and bold curves denote validation error of the center crops. Left: plain networks of 18 and 34 layers. Right: ResNets of 18 and 34 layers. In this plot, the residual networks have no extra parameter compared to their plain counterparts.

The learning rate starts from 0.1 and is divided by 10 when the error plateaus.

So Why is deeper plain model hard to train?

- We argue that this optimization difficulty is *unlikely* to be caused by vanishing gradients. These plain networks are trained with BN, which ensures forward propagated signals to have non-zero variances. We also verify that the backward propagated gradients exhibit healthy norms with BN. So neither forward nor backward signals vanish.
- We conjecture that the deep plain nets may have exponentially low convergence rates, which impact the reducing of the training error3. The reason for such optimization difficulties will be studied in the future.

Going Deeper with Bottleneck Architecture



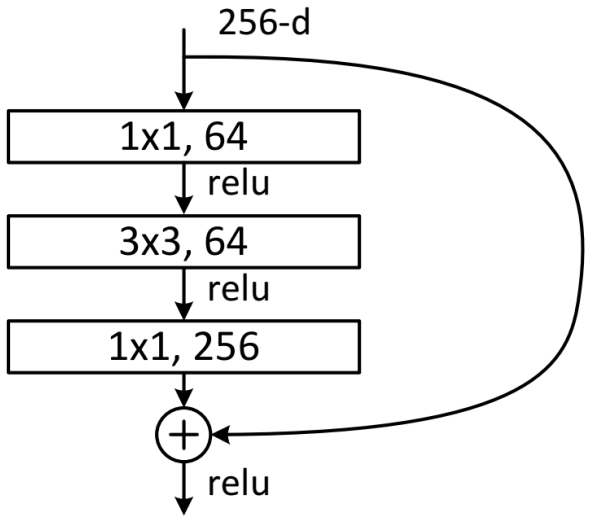


Figure 5. A deeper residual function $\mathcal{F}$ for ImageNet. Left: a building block (on 56×56 feature maps) as in Fig. 3 for ResNet-34. Right: a "bottleneck" building block for ResNet-50/101/152.

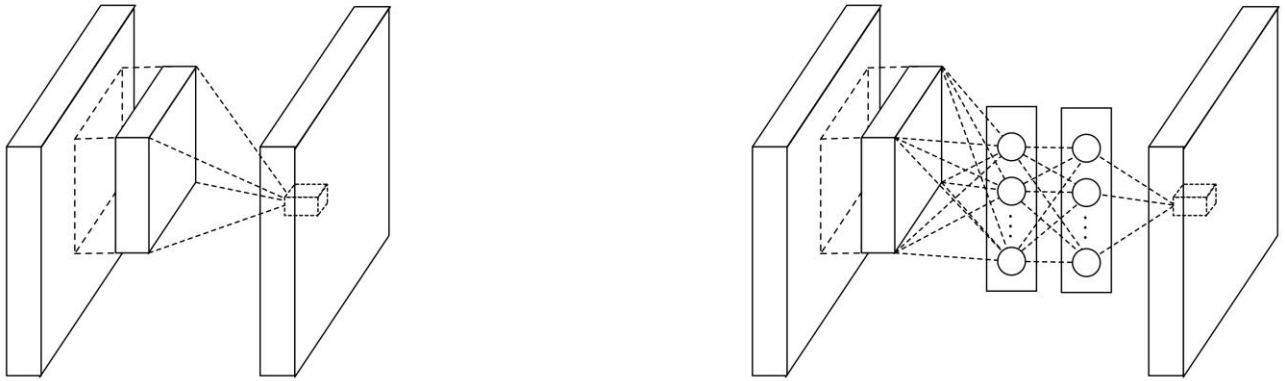
Network in Network

- The convolution filter in CNN is a generalized linear model (GLM) for the underlying data patch, and we argue that the level of abstraction, that the feature is invariant to the variants of the same concept, is low with GLM. Replacing the GLM with a more potent nonlinear function approximator can enhance the abstraction ability.
- Conventional CNN implicitly makes the assumption that the latent concepts are linearly separable. However, the data for the same concept often live on a nonlinear manifold, therefore the representations that capture these concepts are generally highly nonlinear function of the input.
- In NIN, the GLM is replaced with a "micro network" structure which is a general nonlinear function approximator.

Lin et al., 2013, Network in Network, https://arxiv.org/abs/1312.4400v3

Network in Network, mlpconv, 1*1 conv

Multi-layer Perceptron Convolution Layer



(a) Linear convolution layer

(b) Mlpconv layer

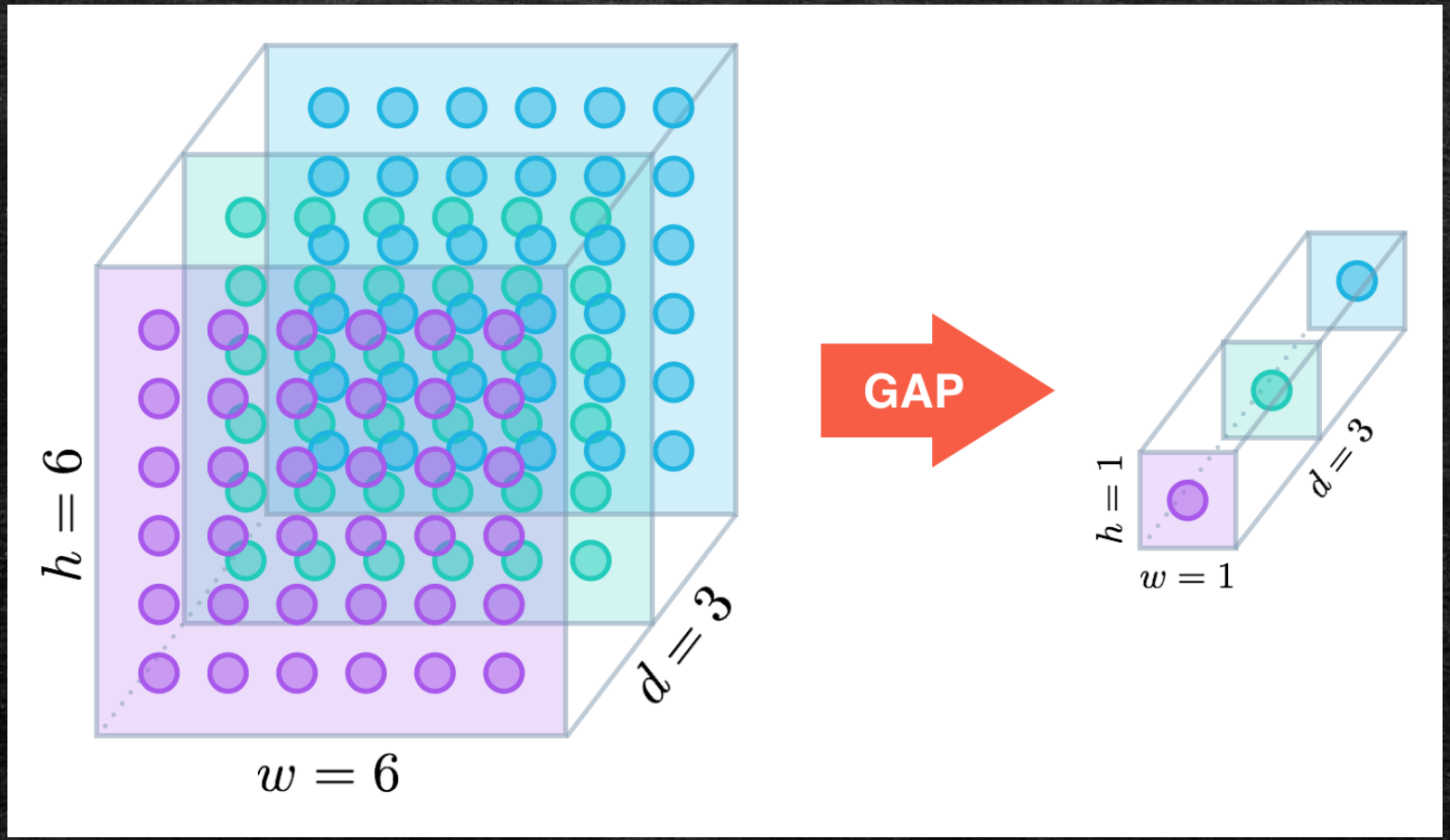
Figure 1: Comparison of linear convolution layer and mlpconv layer. The linear convolution layer includes a linear filter while the mlpconv layer includes a micro network (we choose the multilayer perceptron in this paper). Both layers map the local receptive field to a confidence value of the latent concept.

$$
\begin{aligned}
f^1_{i,j,k_1} &= \max({w^1_{k_1}}^T x_{i,j} + b_{k_1}, 0). \\
&\vdots \\
f^n_{i,j,k_n} &= \max({w^n_{k_n}}^T f^{n-1}_{i,j} + b_{k_n}, 0).
\end{aligned}
$$

3 layer 5*5 mlpconv is mathematically equivalent to 5*5 conv followed by two 1*1 conv

Lin et al., 2013, Network in Network, https://arxiv.org/abs/1312.4400v3

Global Average Pooling



Lin et al., 2013, Network in Network, https://arxiv.org/abs/1312.4400v3

Global Average Pooling

- In traditional CNN, it is difficult to interpret how the category level information from the objective cost layer is passed back to the previous convolution layer due to the fully connected layers which act as a black box in between. In contrast, global average pooling is more meaningful and interpretable as it enforces correspondance between feature maps and categories, which is made possible by a stronger local modeling using the micro network.

Lin et al., 2013, Network in Network, https://arxiv.org/abs/1312.4400v3

Global Average Pooling (GAP)

- In traditional CNN, it is difficult to interpret how the category level information from the objective cost layer is passed back to the previous convolution layer due to the fully connected layers which act as a black box in between.
- In contrast, GAP is more meaningful and interpretable as it enforces correspondance between feature maps and categories, which is made possible by a stronger local modeling using the micro network.
- Feature maps can be easily interpreted as categories confidence maps.
- There is no parameter to optimize in the GAP, thus overfitting is avoided at this layer.
- GAP sums out the spatial information, thus it is more robust to spatial translations of the input.

Lin et al., 2013, Network in Network, https://arxiv.org/abs/1312.4400v3

Architectures of ResNets

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

Table 1. Architectures for ImageNet. Building blocks are shown in brackets (see also Fig. 5), with the numbers of blocks stacked. Down-sampling is performed by conv3_1, conv4_1, and conv5_1 with a stride of 2.

VGG-16/19 nets : 15.3/19.6 billion FLOPs

Comparison Between Networks

method	top-1 err.	top-5 err.
VGG [41] (ILSVRC'14)	-	8.43†
GoogLeNet [44] (ILSVRC'14)	-	7.89
VGG [41] (v5)	24.4	7.1
PReLU-net [13]	21.59	5.71
BN-inception [16]	21.99	5.81
ResNet-34 B	21.84	5.71
ResNet-34 C	21.53	5.60
ResNet-50	20.74	5.25
ResNet-101	19.87	4.60
ResNet-152	**19.38**	**4.49**

Table 4. Error rates (%) of **single-model** results on the ImageNet validation set (except † reported on the test set).