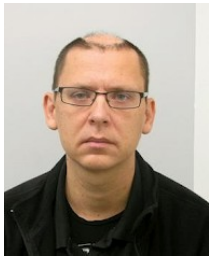
Going deeper with convolutions

GoogLeNet

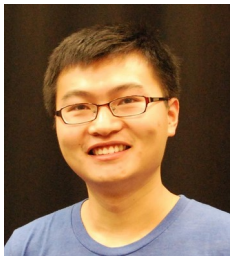
BIL722 Advanced Vision - Presentation

Mehmet Günel

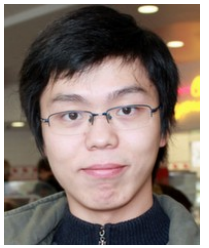
Team



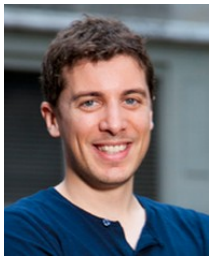
Christian Szegedy,
Google



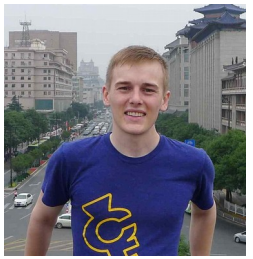
Wei Liu,
UNC



Yangqing Jia,
Google



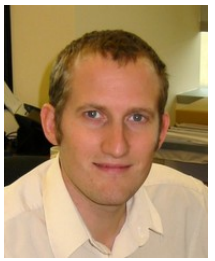
Pierre Sermanet,
Google



Scott Reed,
University of Michigan

Dragomir Anguelov,
Google

Dumitru Erhan,
Google



Vincent Vanhoucke,
Google

Andrew Rabinovich,
Google

Basics

- What is ILSVRC14?

  - ImageNet Large-Scale Visual Recognition Challenge 2014

- What is ImageNet?

  - WordNet hierarchy, concept = "synonym set" or "synset".

  - More than 100,000 synsets in WordNet, on average 1000 images to illustrate each synset

- What are Google Inception and GoogLeNet?

  - 

Overview of the GoogleNet

- A deep convolutional neural network architecture
- Classification and detection for ILSVRC14
- Improved utilization of the computing resources inside the network

  while increasing size, both depth and width
- 12x fewer parameters than the winning architecture of Krizhevsky
- Significantly more accurate than state of the art
- 22 layers deep when counting only layers with parameters
- The overall number of layers (independent building blocks) used for the construction of the network is about 100

What is the Problem?

- Aim:
  - To improve the performance of classification and detection

- Restrictions:
  - Usage of CNN
  - Able to train with smaller dataset
  - Limited computational power and memory usage

How to improve classification and detection rates?

- Straightforward approach;

  Jut increase the size of network in both direction !

BUT!!!

Straightforward approach, challenge 1

- Larger number of parameters → Requires bigger data;

  Otherwise overfit! High quality training sets can be tricky and expensive...

(a) Siberian husky

(b) Eskimo dog

Straightforward approach, challenge 2

- Dramatically increased use of computational resources!
- A simple example:
  - If two convolutional layers are chained, any uniform increase in the number of their filters results in a quadratic increase of computation

What is their approach?

- Moving from fully connected to sparsely connected architectures, even inside the convolutions

Handicap of the sparse approach

- Todays computing infrastructures are very inefficient when it comes to numerical calculation on non-uniform sparse data structures
- The gap is widened even further by the use of steadily improving, highly tuned, numerical libraries that allow for extremely fast dense matrix multiplication, exploiting the minute details of the underlying CPU or GPU hardware
- Also, non-uniform sparse models require more sophisticated engineering and computing infrastructure
- Even people go back to fully connected approach!

Their Solution

- An architecture that makes use of the extra sparsity, even at filter level, as suggested by the theory, but exploits our current hardware by utilizing computations on dense matrices

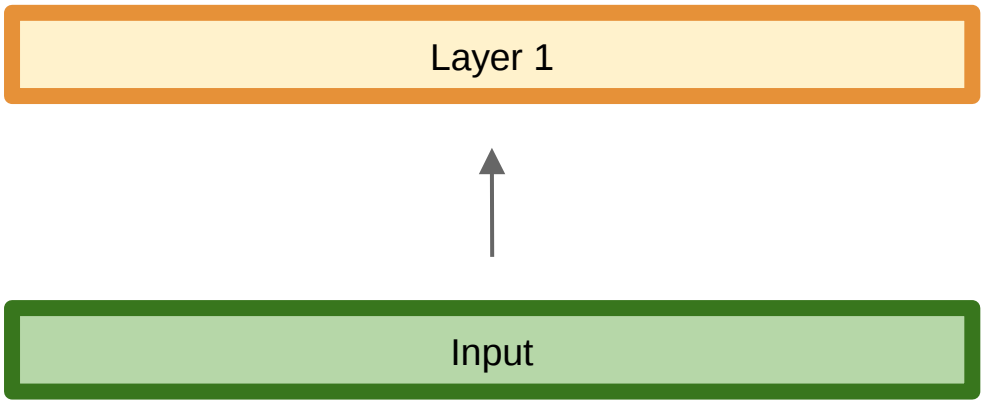
- Clustering sparse matrices into relatively dense submatrices tends to give state of the art practical performance for sparse matrix multiplication

Their motivation

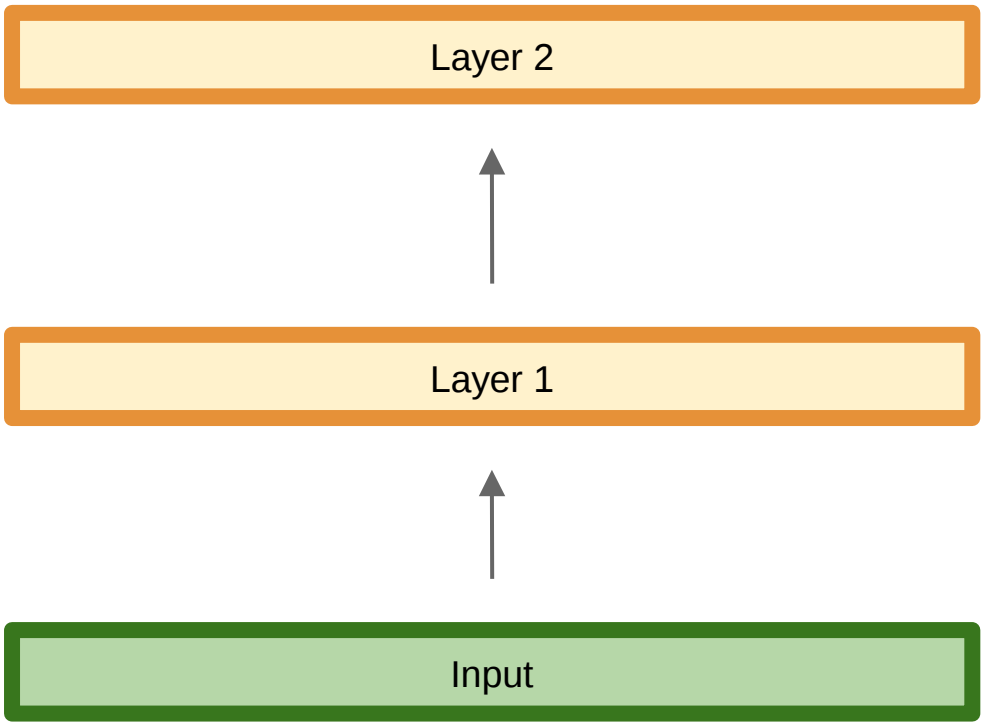
- Multi-scale processing namely synergy of deep architectures and classical computer vision, like the R-CNN algorithm by Girshick
- If the probability distribution of the data-set is representable by a large, very sparse deep neural network, then the optimal network topology can be constructed layer by layer by analyzing the correlation statistics of the activations of the last layer and clustering neurons with highly correlated outputs
- Hebbian principle: neurons that fire together, wire together

Hebbian Principle

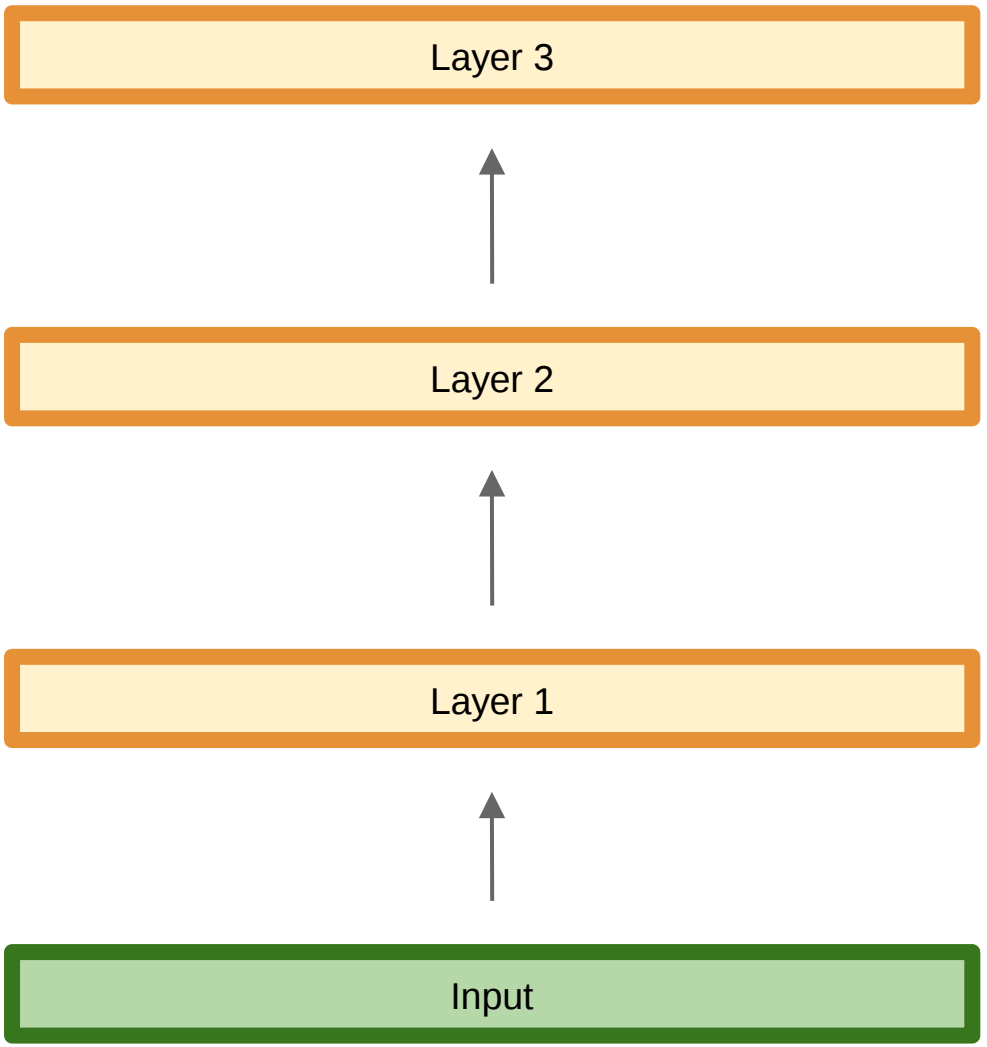
Cluster according activation statistics



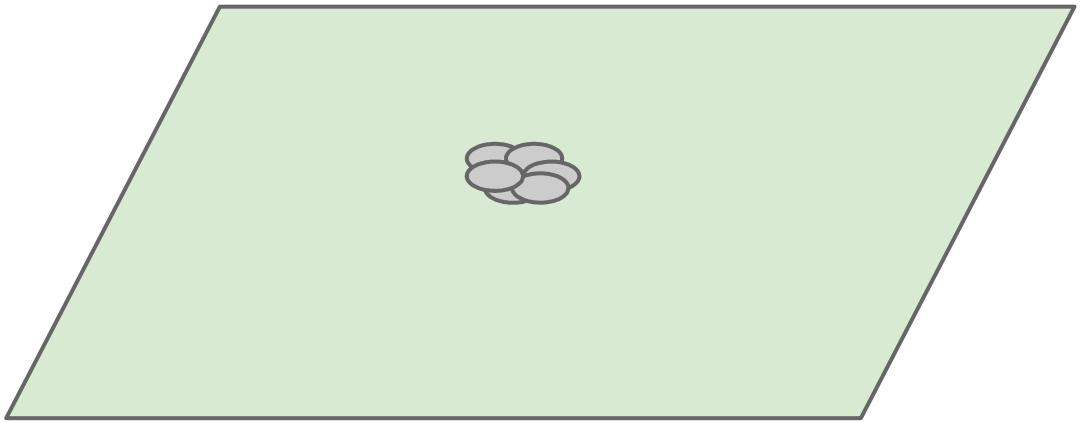
Cluster according correlation statistics



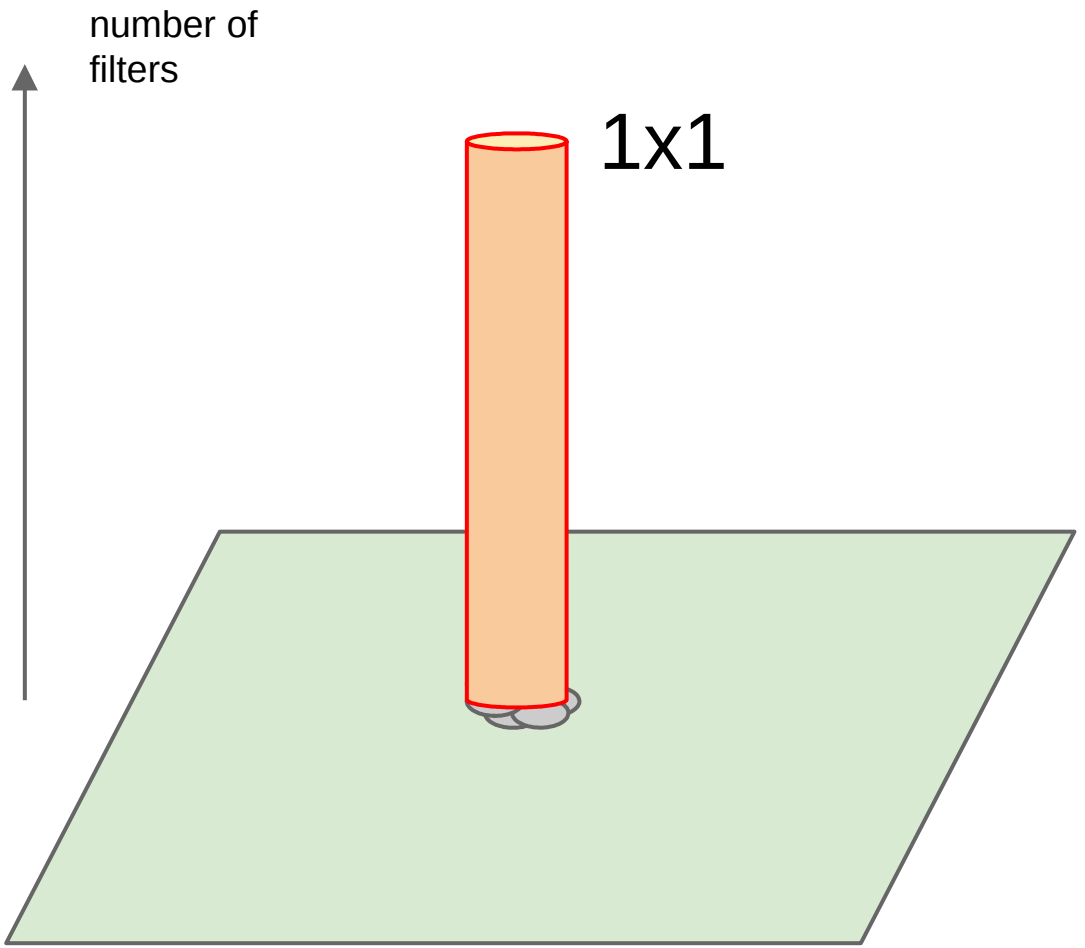
Cluster according correlation statistics



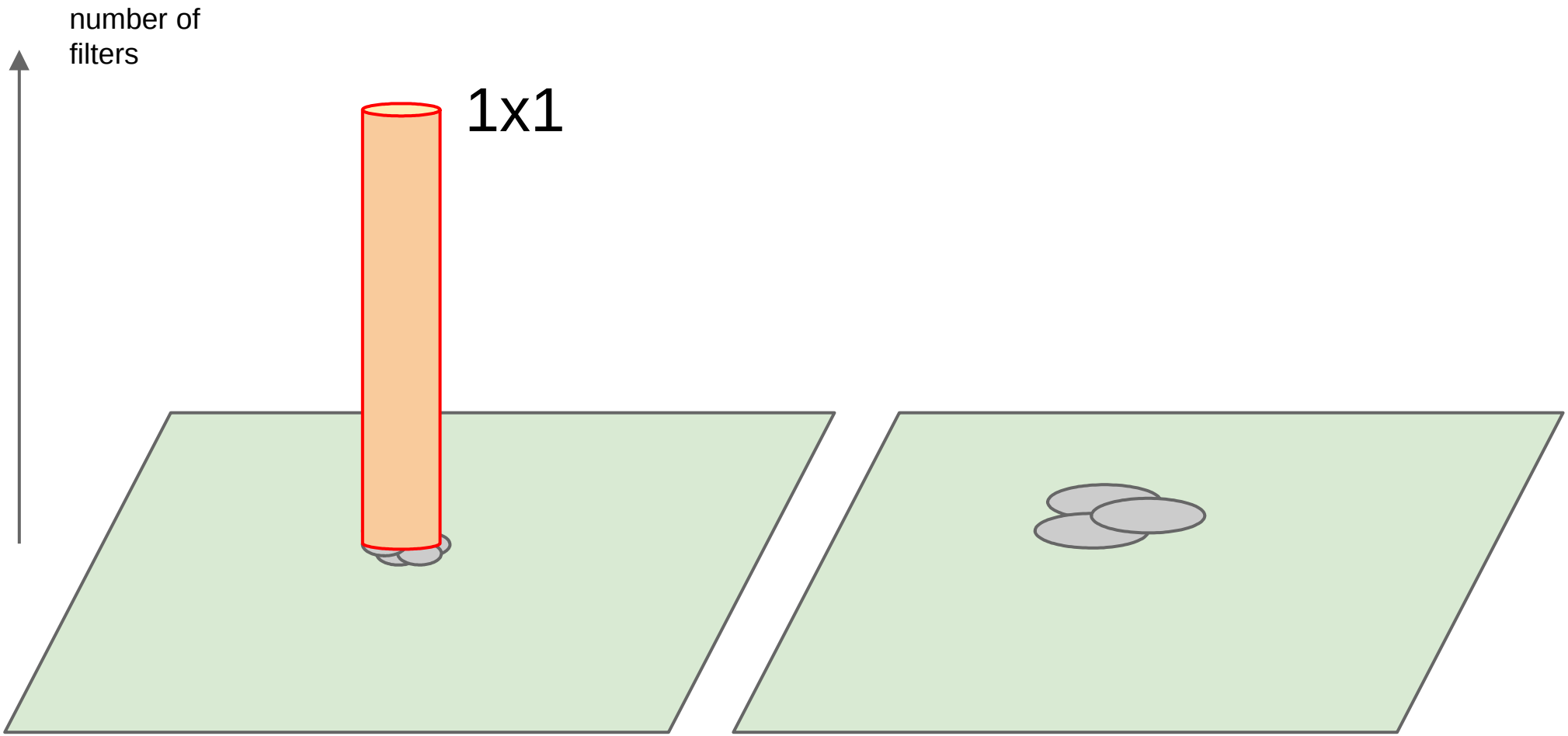
In images, correlations tend to be local



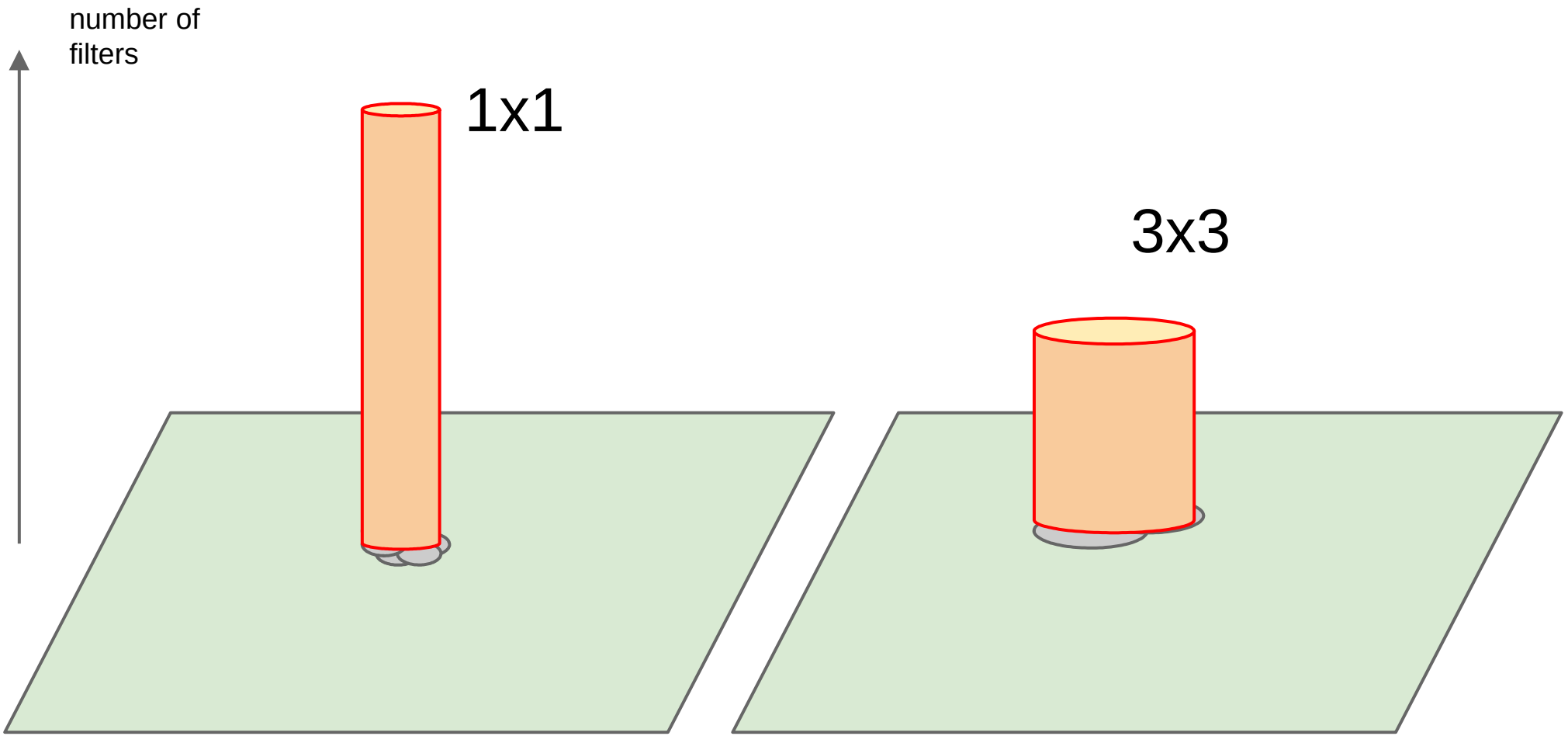
Cover very local clusters by 1x1 convolutions



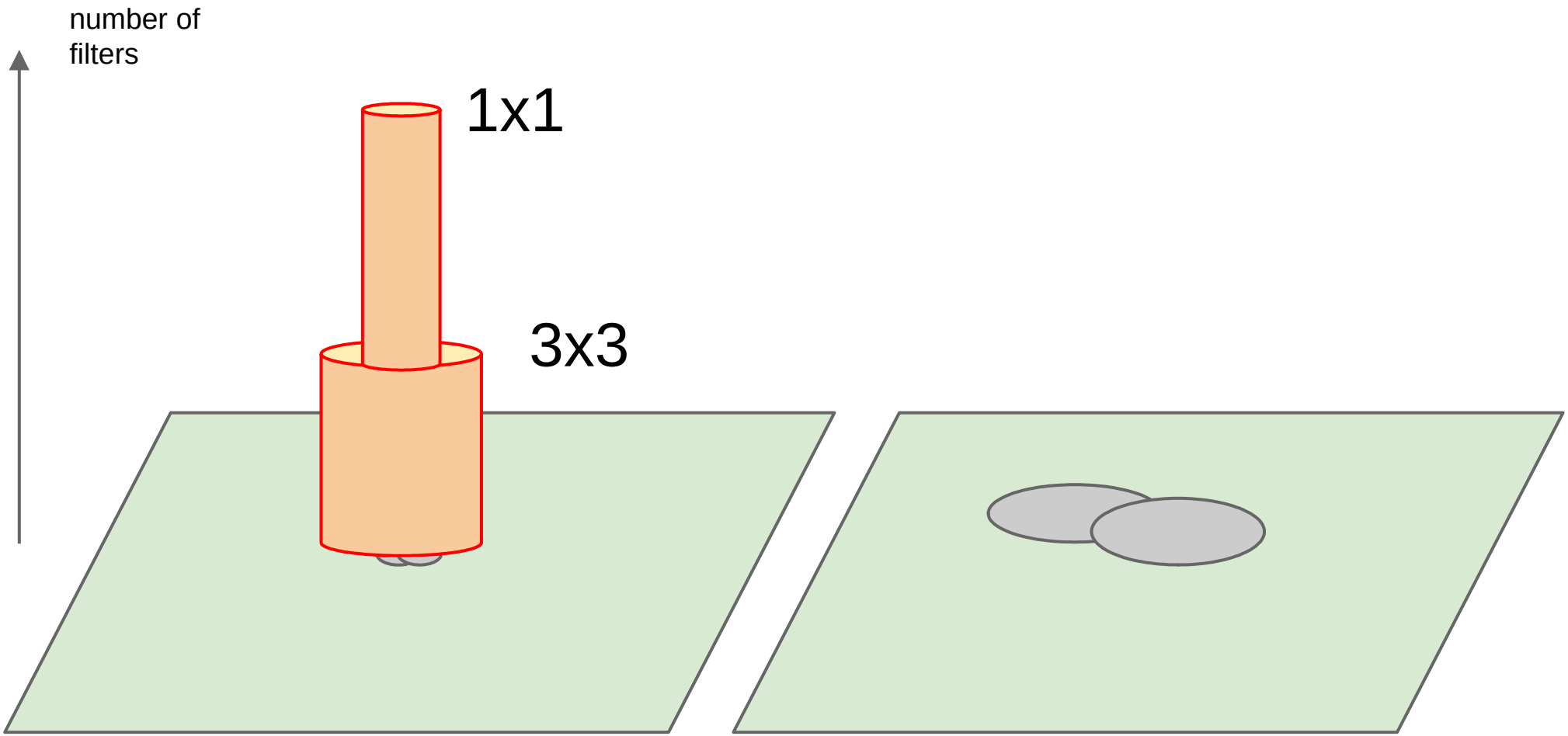
Less spread out correlations



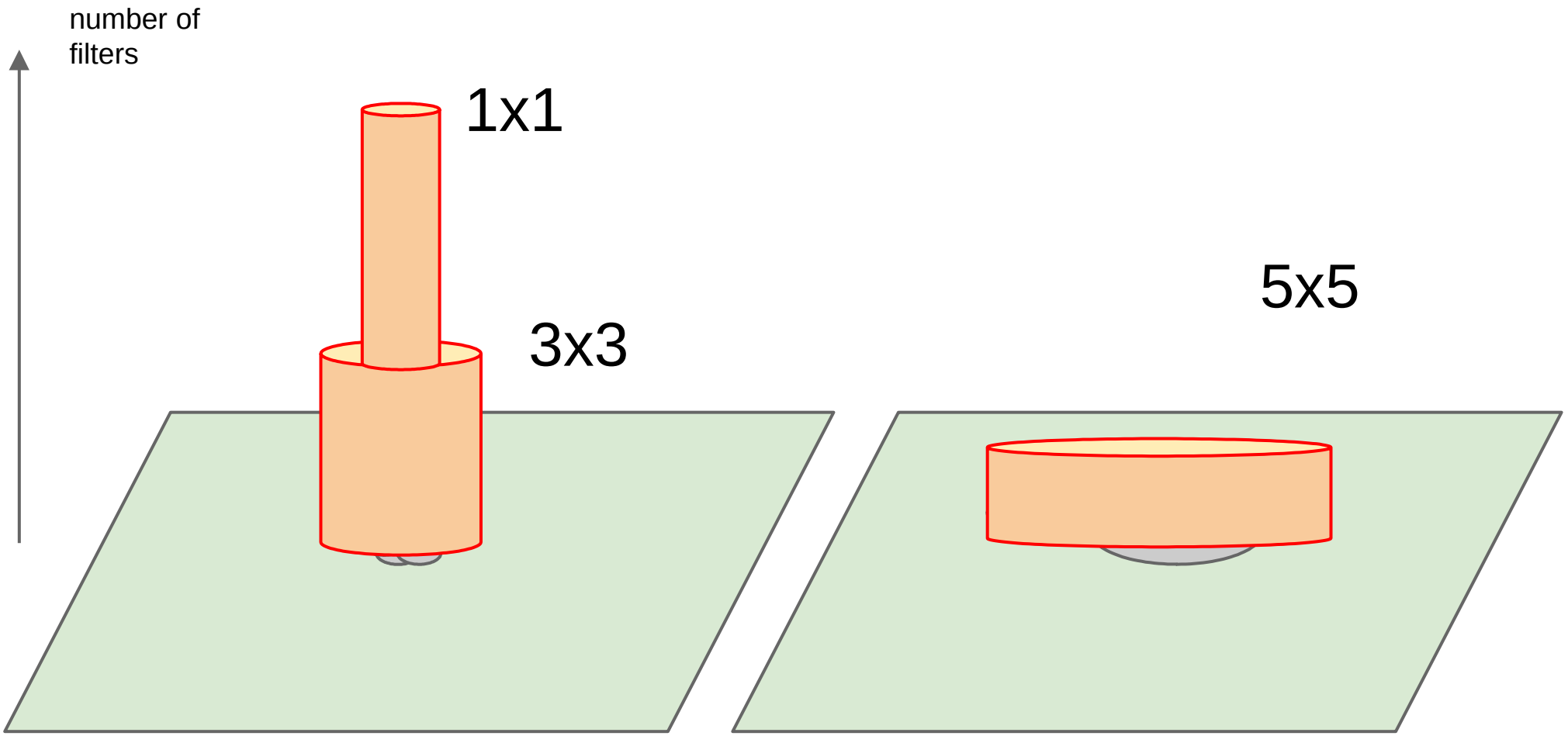
Cover more spread out clusters by 3x3 convolutions



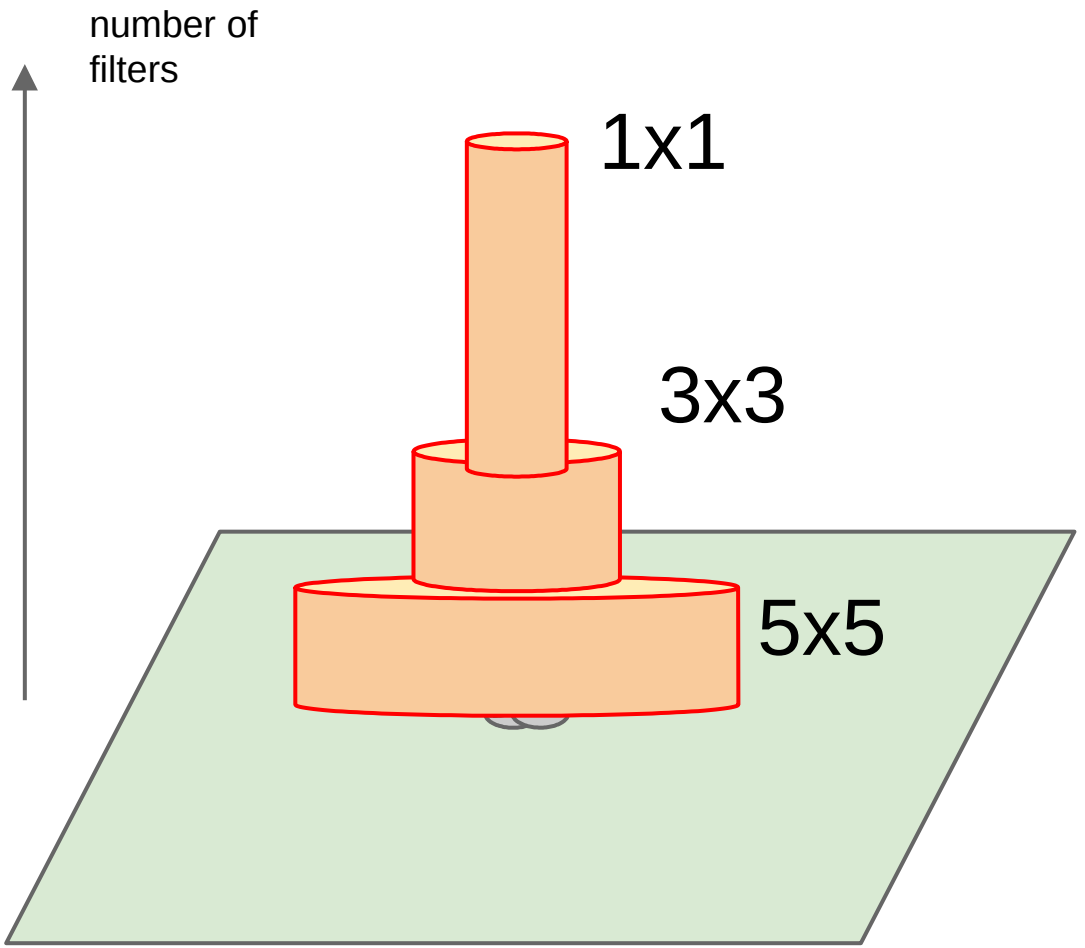
Cover more spread out clusters by 5x5 convolutions



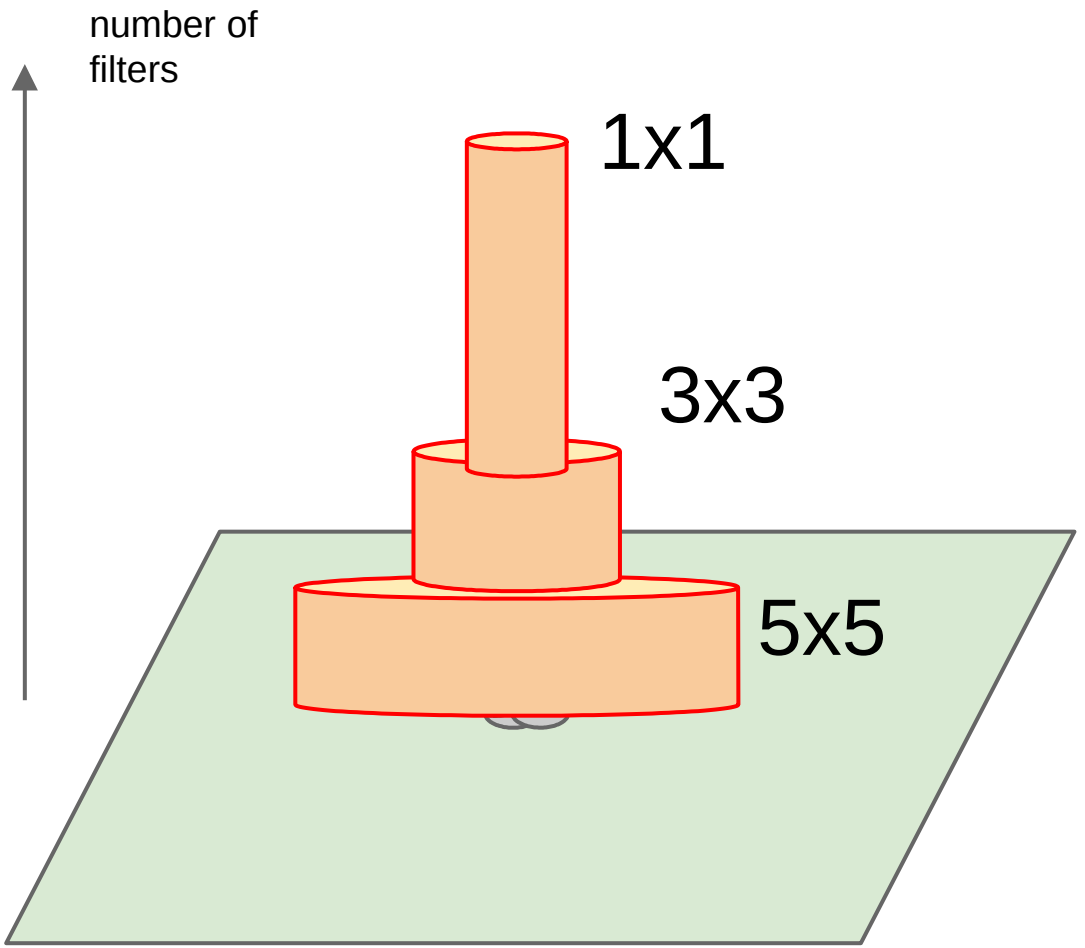
Cover more spread out clusters by 5x5 convolutions

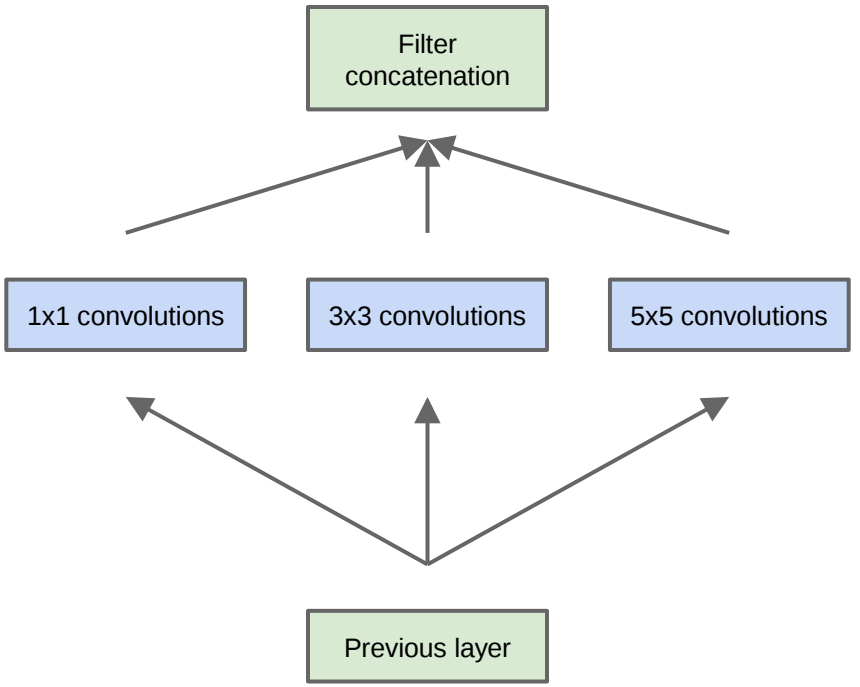


A heterogeneous set of convolutions

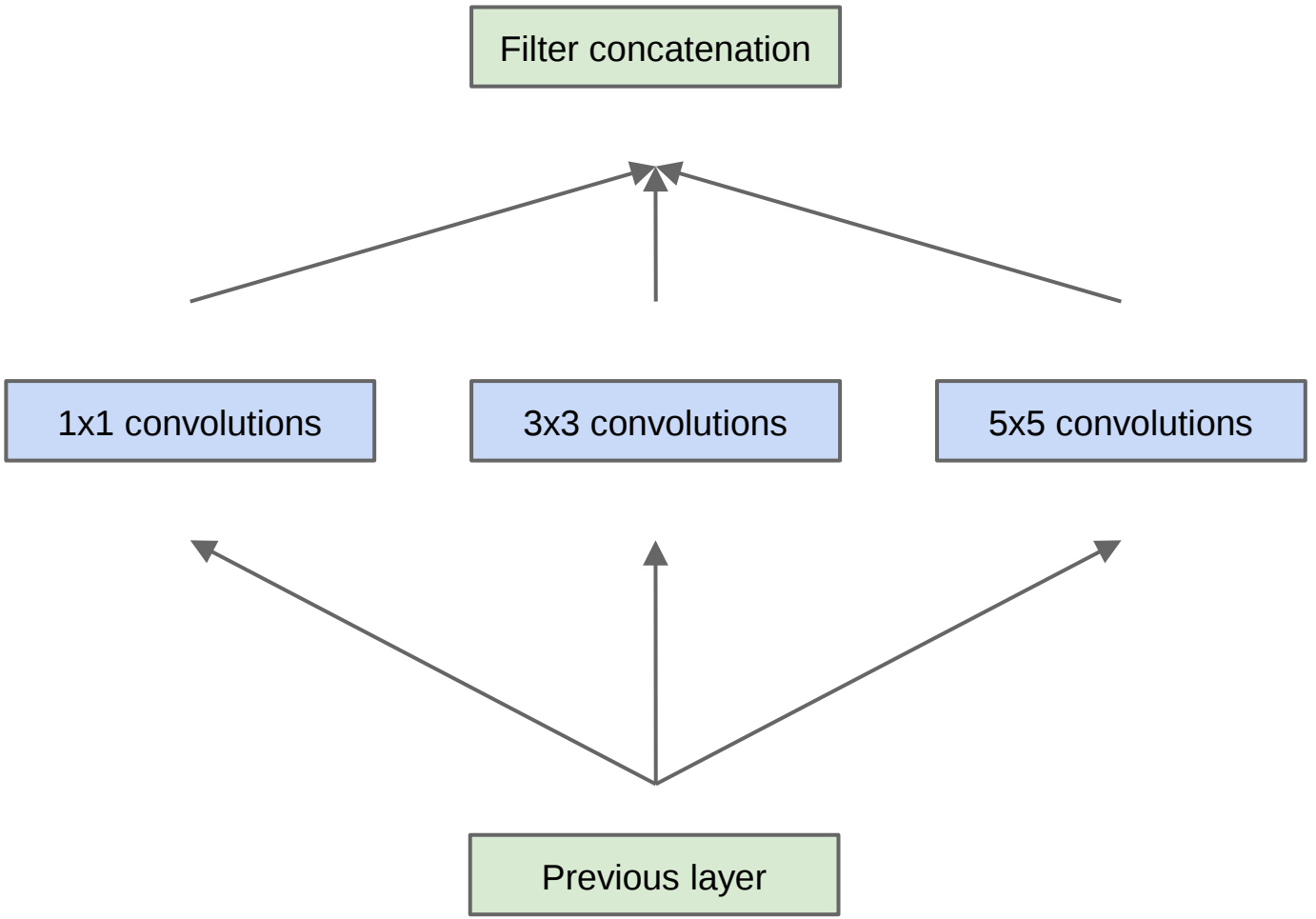


Schematic view (naive version)

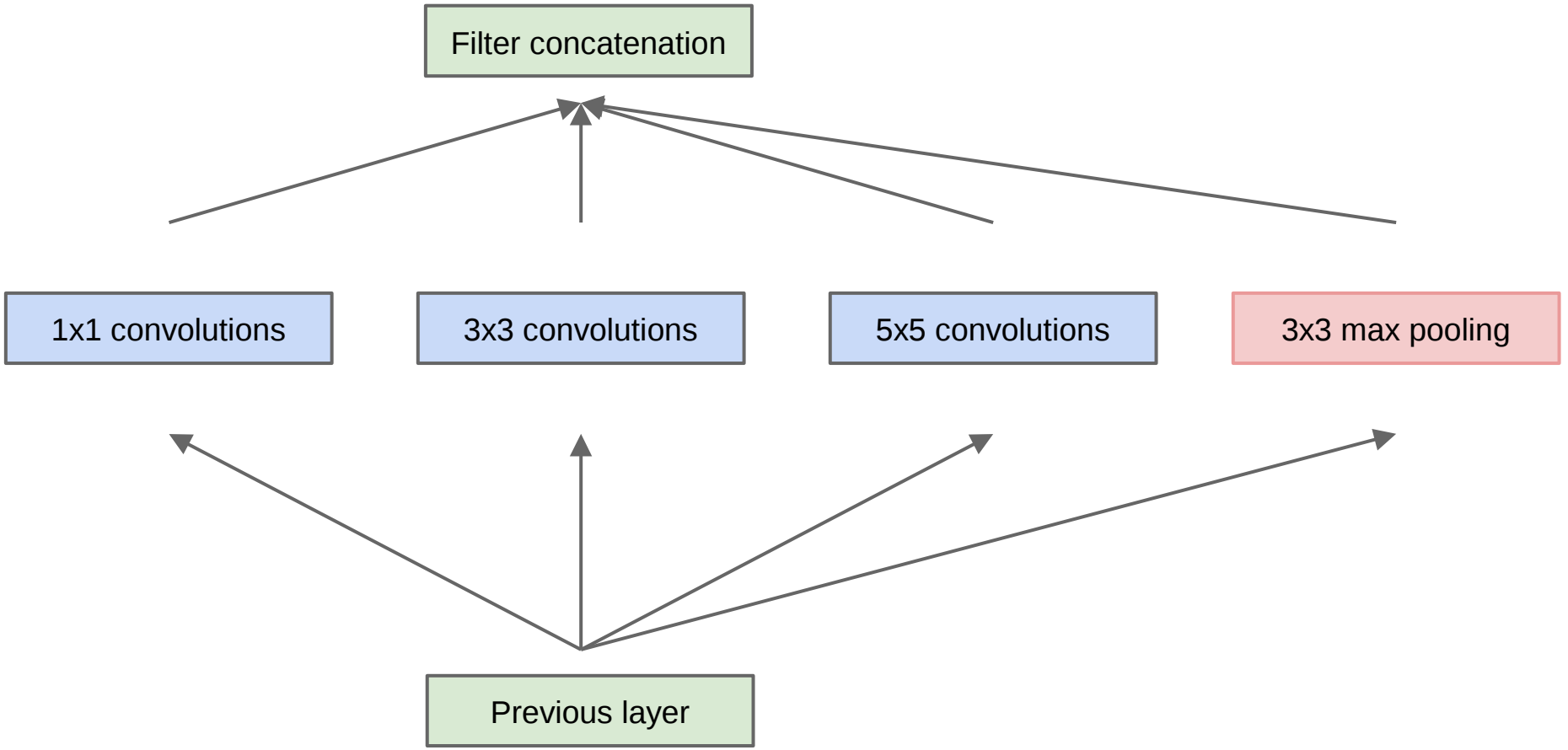




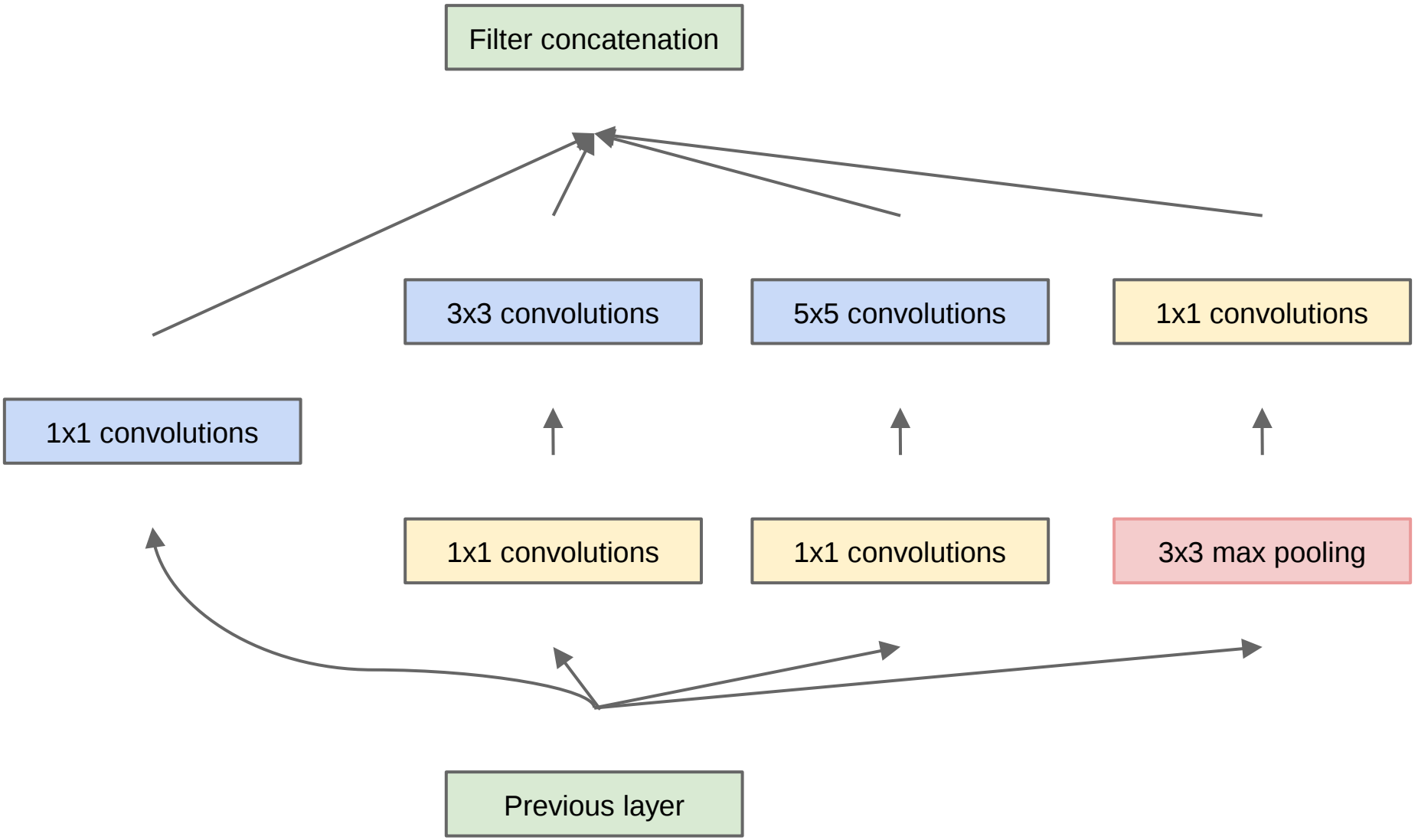
Naive idea


Filter concatenation
1x1 convolutions
3x3 convolutions
5x5 convolutions
Previous layer

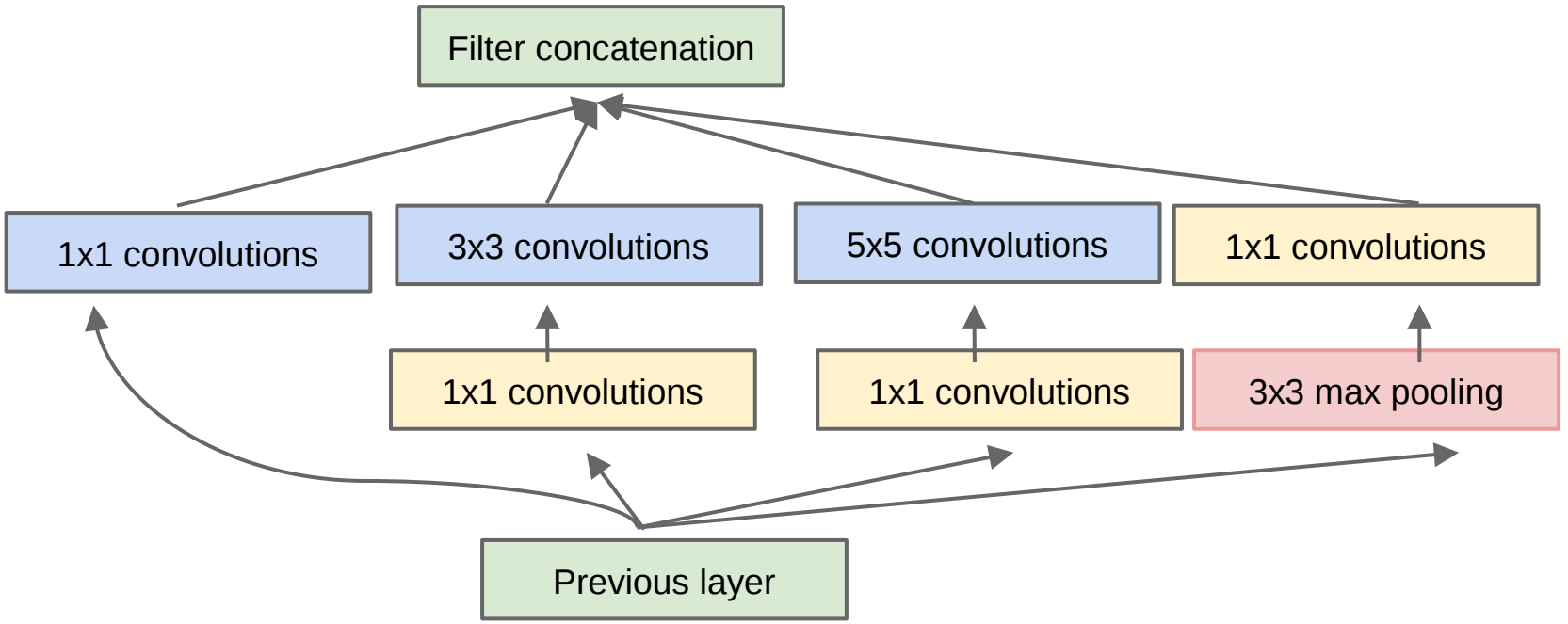
Naive idea (**does not work!**)



Inception module



Inception module



- 1×1 convolutions are used to compute reductions before the expensive 3×3 and 5×5 convolutions.
- Besides being used as reductions, they also include the use of rectified linear activation which makes them dual-purpose

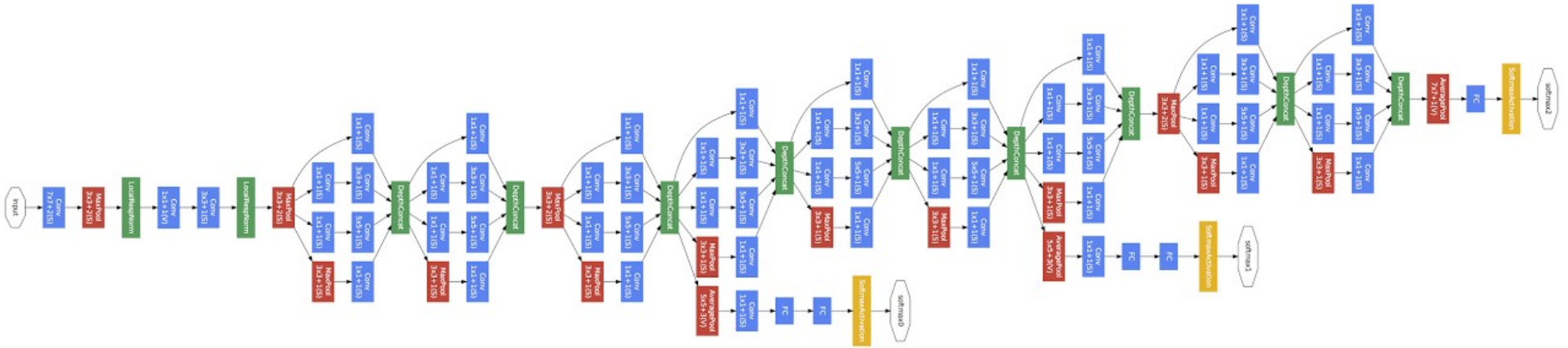
How these 1x1 convolutions work?

- Receptive field
- Not dimensionality reduction in space, but can dimensionality reduction in channel
- ReLU functionality

Solution Details

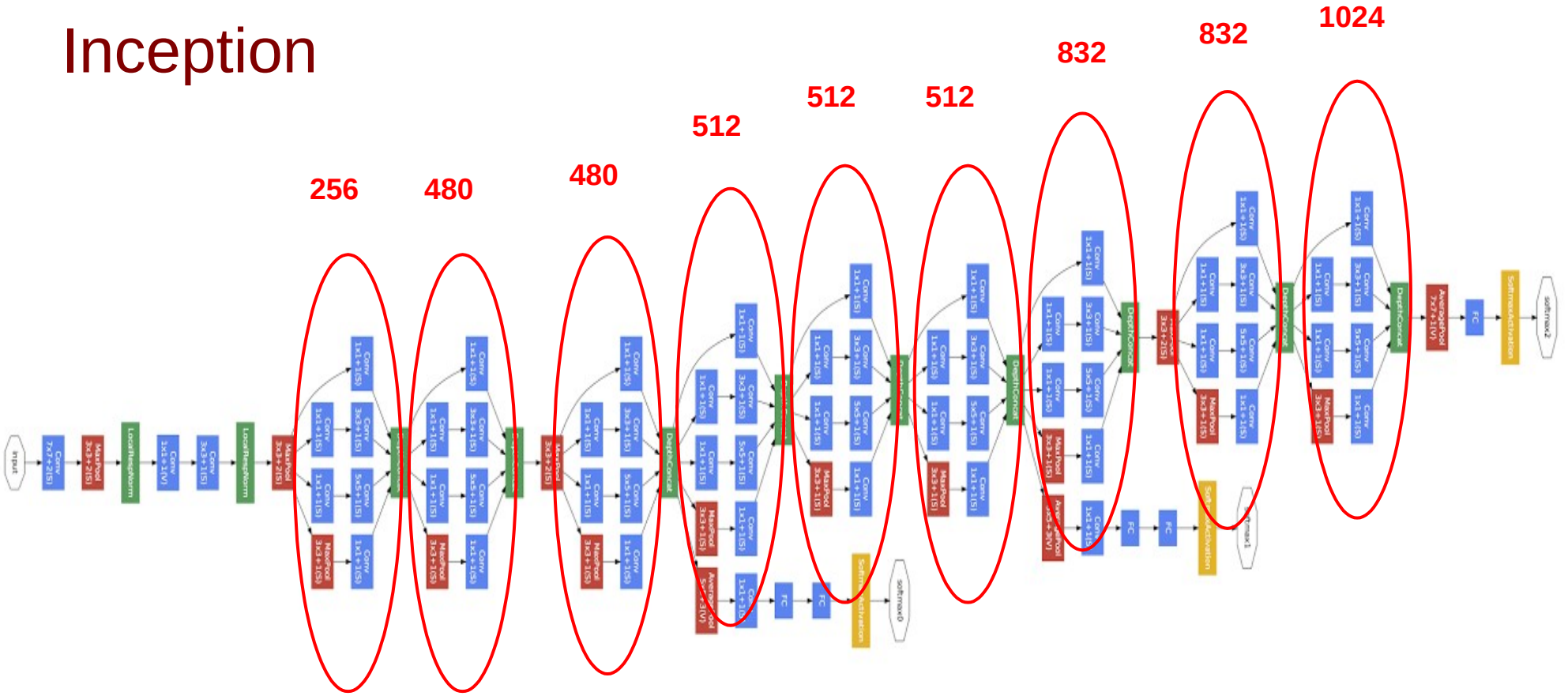
- Optimal local sparse structure in a convolutional vision network can be approximated and covered by readily available dense components
- Find the optimal local construction and repeat it spatially

GoogLeNet



Convolution
Pooling
Softmax
Other

Inception



Width of **inception modules** ranges from 256 filters (in early modules) to 1024 in top inception modules.

Can remove fully connected layers on top completely

Number of parameters is reduced to 5 million

Computional cost is increased by less than 2X compared to Krizhevsky's network. (<1.5Bn operations/evaluation)

Auxiliary classifiers

- Encourage discrimination in the lower stages in the classifier
- Increase the gradient signal that gets propagated back
- Provide additional regularization

Auxiliary classifiers

- An average pooling layer with 5x5 filter size and stride 3, resulting in an 4x4x512 output or the (4a), and 4x4x528 for the (4d) stage.

- A 1x1 convolution with 128 filters for dimension reduction and rectified linear activation.

- A fully connected layer with 1024 units and rectified linear activation.

- A dropout layer with 70% ratio of dropped outputs.

- A linear layer with softmax loss as the classifier (predicting the same 1000 classes as the main classifier, but removed at inference time)

Training

- CPU based implementation
- Asynchronous stochastic gradient descent with 0.9 momentum
- Fixed learning rate schedule (decreasing the learning rate by 4% every 8 epochs)
- Polyak averaging at inference time
- Sampling of various sized patches of the image whose size is distributed evenly between 8% and 100%
- Photometric distortions to combat overfitting
- Random interpolation methods (bilinear, area, nearest neighbor and cubic, with equal probability) for resizing

Classification Experimental Setup and Results

- 1000 leaf-node categories
- About 1.2 million images for training. 50,000 for validation and 100,000 images for testing
- Each image is associated with one ground truth category
- Performance is measured based on the highest scoring classifier predictions

Classification Experimental Setup and Results

- Main metrics are;
  - *top-1 accuracy rate*: compares the ground truth against the first predicted class
  - *top-5 error rate*: compares the ground truth against the first 5 predicted classes (image is correctly classified if the ground truth is among the top-5, regardless of its rank in them)

  The challenge uses the top-5 error rate for ranking purposes

Classification Experimental Setup and Results

- Tricks and techniques;
  - *Ensemble*: 7 versions of the same GoogLeNet, trained with the same initialization & learning rate. Only differ in sampling methodologies and the random order in which they see input images
  - *Data manipulation*: Agressive cropping, resize the image to 4 scales (256, 288, 320 and 352) and take squares of these resized images. Result is 4×3×6×2 = 144 crops per image
  - *Averaging*: softmax probabilities are averaged over multiple crops and over all the individual classifiers to obtain the final prediction

Classification results on ImageNet

Number of Models	Number of Crops	Computational Cost	Top-5 Error	Compared to Base
1	1 (center crop)	1x	10.07%	-
1	10*	10x	9.15%	-0.92%
1	144 (Our approach)	144x	7.89%	-2.18%
7	1 (center crop)	7x	8.09%	-1.98%
7	10*	70x	7.62%	-2.45%
7	144 (Our approach)	1008x	6.67%	-3.41%

*Cropping by [Krizhevsky et al 2014]

Classification results on ImageNet

Team	Year	Place	Error (top-5)	Uses external data
SuperVision	2012	-	16.4%	no
SuperVision	2012	1st	15.3%	ImageNet 22k
Clarifai	2013	-	11.7%	no
Clarifai	2013	1st	11.2%	ImageNet 22k
MSRA	2014	3rd	7.35%	no
VGG	2014	2nd	7.32%	no
GoogLeNet	2014	1st	6.67%	no

Detection Experimental Setup and Results

- Produce bounding boxes around objects in images
- 200 possible classes.
- Detected objects count as correct if they match the class of the groundtruth and their bounding boxes overlap by at least 50%
- Extraneous detections count as false positives and are penalized
- Each image may contain many objects or none, and their scale may vary from large to tiny

Detection Experimental Setup and Results

- Tricks and techniques;
  - Similar to R-CNN, Inception model as the region classifier
  - Selective Search approach combined with multi-box predictions
  - Superpixel size was increased by 2x in order to decrease false positives
  - Ensemble of 6 ConvNets

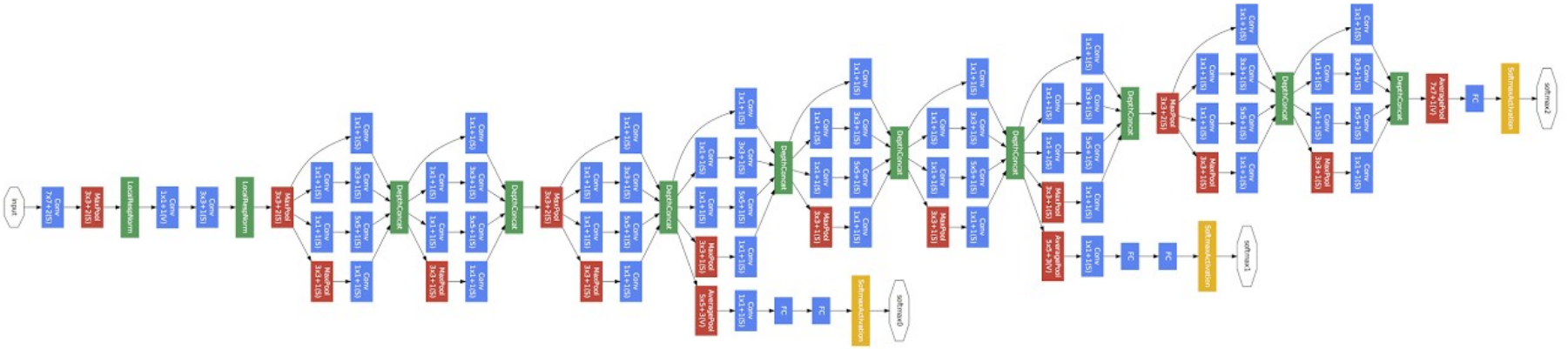
Detection results without ensembling

Team	mAP	external data	contextual model	bounding-box regression
Trimps-Soushen	31.6%	ILSVRC12 Classification	no	?
Berkeley Vision	34.5%	ILSVRC12 Classification	no	yes
UvA-Euvision	35.4%	ILSVRC12 Classification	**?**	**?**
CUHK DeepID-Net2	37.7%	ILSVRC12 Classification+ Localization	no	?
GoogLeNet	38.0%	ILSVRC12 Classification	no	no
Deep Insight	40.2%	ILSVRC12 Classification	yes	yes

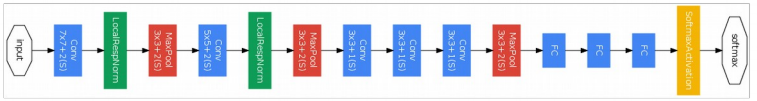
Final Detection Results

Team	Year	Place	mAP	external data	ensemble	contextual model	approach
UvA-Euvision	2013	1st	22.6%	none	?	yes	Fisher vectors
Deep Insight	2014	3rd	40.5%	ILSVRC12 Classification + Localization	3 models	yes	ConvNet
CUHK DeepID-Net	2014	2nd	40.7%	ILSVRC12 Classification + Localization	?	no	ConvNet
GoogLeNet	2014	1st	43.9%	ILSVRC12 Classification	6 models	no	ConvNet

GoogLeNet vs State of the art



GoogLeNet



Zeiler-Fergus Architecture (1 tower)

Convolution
Pooling
Softmax
Other

Classification failure cases

Groundtruth: **????**

Classification failure cases

Groundtruth: **coffee mug**

Classification failure cases

Groundtruth: **coffee mug**

GoogLeNet:

- **table lamp**
- **lamp shade**
- **printer**
- **projector**
- **desktop computer**

Classification failure cases

Groundtruth: **???**

Classification failure cases

Groundtruth: **Police car**

Classification failure cases

Groundtruth: **Police car**

GoogLeNet:

- **laptop**
- **hair drier**
- **binocular**
- **ATM machine**
- **seat belt**

Classification failure cases

Groundtruth: **???**

Classification failure cases

Groundtruth: **hay**

Classification failure cases

Groundtruth: **hay**

GoogLeNet:

- **sorrel (horse)**
- **hartebeest**
- **Arabian camel**
- **warthog**
- **gaselle**

Cons and doubts

- One must be cautious though: although the proposed architecture has become a success for computer vision, it is still questionable whether its quality can be attributed to the guiding principles that have lead to its construction
- No specific training methodology

Conclusion and future work

- Approximating the expected optimal sparse structure by readily available dense building blocks is a viable method for improving neural networks for computer vision
- Low computational requirements
- Thus, moving to sparser architectures is feasible and useful idea in general
- Future work: creating sparser and more refined structures in automated ways

Thanks

Questions?