Going Deeper With Convolutions

Szegedy et al.

Sourced from:
Going Deeper with Convolutions – Szegedy et al.
Representation Learning: A Review and New Perspectives – Bengio et al.
Network in Network – Lin et al.
http://www.robots.ox.ac.uk/~vgg/practicals/cnn/

Mukul Sati
03/31/2016

Representation learning via CNNs

- Ideas behind deep CNN design:
  - Learn weights w_i such that $f:x->f_L(\ldots f_2(f_1(x;w_1);w_2)\ldots),w_L)$ transforms input points to a “good” space, for known f_i's.
  - “Good” space - disentangles factors of variation.
  - Each layer composed of “neurons” as units with:
    - Local receptive field - Each unit fires or not based on input values in a local neighborhood.
    - Spatial invariance of neurons - All units in a layer share the same weights.
  - Pooling – Build in some input translation invariance.

Some terms

- **Abstract concept** – concept that we wish to capture from input data.
  - Looking at the entire image – abstract concept – dog, cat, etc.
  - Looking at a local patch – abstract concept – 45 degree edge, green textured blob, etc.
- Note:
  - Abstract concepts are invariant to local changes of input.
  - Thus, “good” space mappings are highly non-linear mappings of the input data.

- **(More) Abstract feature** – transformed data that is (more) removed from the input data (i.e, has no simple interpretation) – can capture abstract concepts.

  Note: From the above note, as abstract concepts correspond to highly non-linear mappings, we need more abstract features to capture them.

Obtaining more abstract features

- Larger number of layers in the deep nets allow for the capture of more abstract features.
  - $f:x->f_L(\ldots f_2(f_1(x;w_1);w_2)\ldots),w_L)$ vs $f:x->f_2(f_1(x;w_1);w_2)$

So, bigger CNNs

- Increase:
  - Number of layers.
  - Number of neurons per layer.

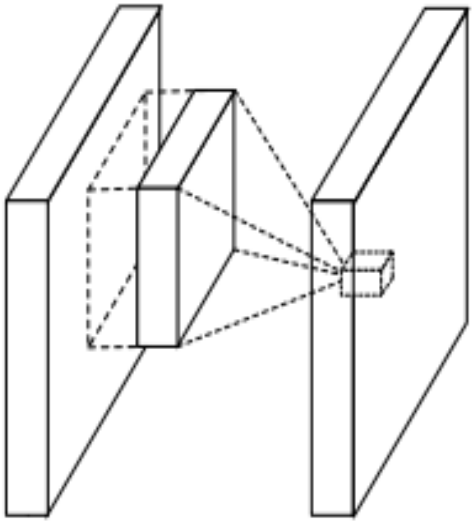
- Impact:
  - Chances of overfitting increase.
  - Computational cost scales poorly with increase in number of learned filters.

Abstract features for local patches?

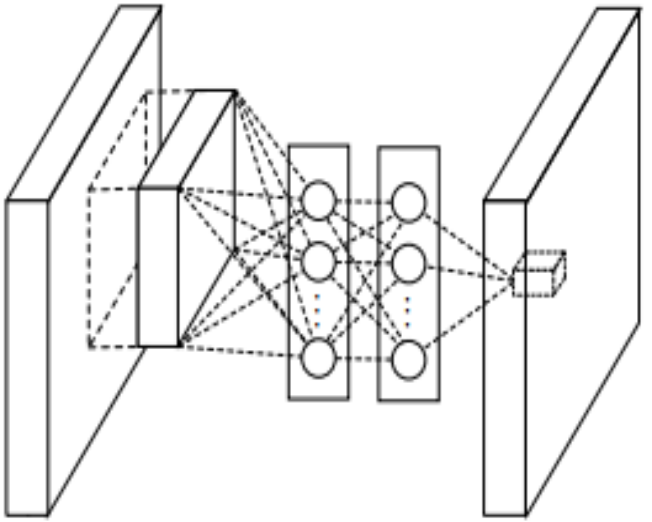
- While abstract features are learned using deeper nets, local patches are obtained just by the composition of a convolution and a non-linear activation.
- These are not abstract enough.
- Not exactly true, but if we think of a single neuron activating for one abstract local concept. Then, the abstraction provided by a convolution(linear) + re-lu(non-linear) is not enough to capture variations of the same local concept
- Under this interpretation, we can interpret the success of CNNs to be due to each CNN layer learning an over-complete set of local filters to capture these variations.
- If more abstract features are captured locally, perhaps we'll need lower number of filters.

Abstract features for local patches

- Replace convolution by multi-layered perceptron (locally).
- Slide this unit for CNNs similar to the linear convolutional layer over the entire image, as we still want the spatial invariance of neurons.



(a) Linear convolution layer



(b) Mlpconv layer

Abstract features for local patches

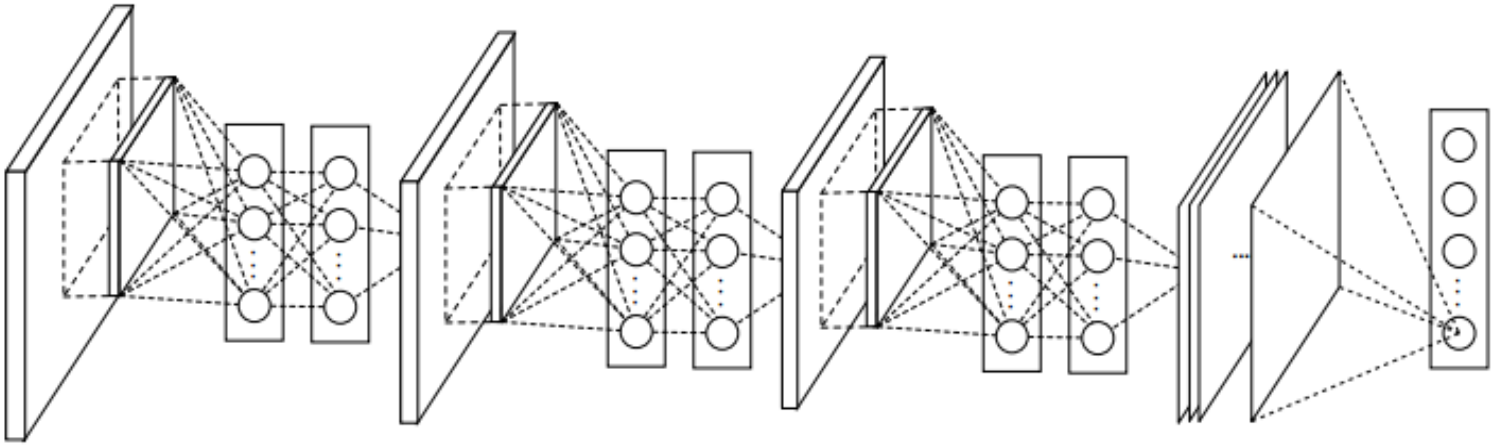
- The multi-layered perceptron layers do the following:

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

- This sequence of compositions can also be understood as pooling across the input feature maps.
- This pooling is equivalent to convolution with a 1X1 convolution kernel.
- So, the action of a MLP can be understood as that of 1X1 convolutional kernel (**important takeaway**).

Global Average Pooling

- Using abstract local features, it is possible to interpret each neuron as firing for one abstract concept (as richer local abstractions can be captured), and thus, the network can be trained this way.
- Specifically, at the final layer, remove the fully connected layer generally present before softmax logit layer. Instead, generate one feature map for each classification category, and directly drive softmax layer with it (take average of each feature map and pass it softmax).
- Note – Reiterate - This was not possible in normal conv layers where less abstract concepts were captured locally, and thus, the final layer needed to be fully connected to allow complex interaction between underlying concepts.
- Average pooling also eases understanding by making each feature map of the final layer directly correspond to a class.



Ideal sparse network – Arora et al.

- If the probability distribution of the dataset is representable by a large, sparse deep neural network, optimal network topology can be constructed by connecting neurons that yield highly correlated outputs (activations) to an upper layer neuron.
  - Neuron's that fire together, wire together.

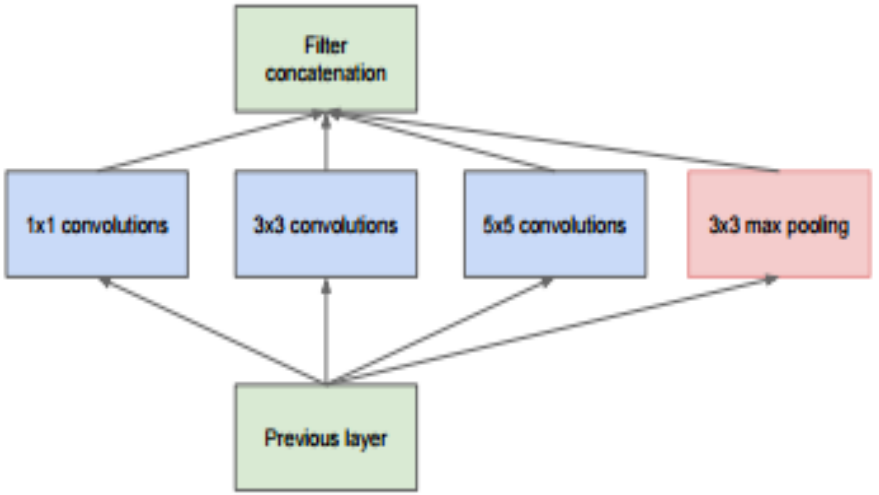
Learning locally optimal sparse network as suggested by Arora et al.

- Non optimized sparse computes with today's computation paradigms.
- Instead, create an architecture that can **cover** all possible locally optimal sparse structures by existing fully connected structure to facilitate use of highly optimized / GPU dense data processing frameworks.
- Then, learn the optimal sparse local structure from data.
- Still want to crudely mimic human vision, so use CNN architecture, just replacing convolution with the learned optimal sparse structure.

Covering optimal sparse local network structures

- In optimal sparse structure, correlated neurons will be wired together to a neuron in a higher layer (Arora et al).
- By spatial coherence of images, these are neurons that look at the same space.
  - Cover these by 1X1 convolutions. (Till now, exactly same as network in network and thus, also interpretable as feature map pooling).
- Also, allow the possibility that a collection of spatially adjacent neurons are wired together.
  - Use set of 3X3 convolutions.
  - Use set of 5X5 convolutions.
- Also, use max-pooling (3X3 with stride of 1) as well (for translation invariance).
- At end, stack up all filter bank activations – note this will require padding of 1, 2 and 1 for the 3X3, 5X5 convs and 3X3 max pool.

Naive inception architecture

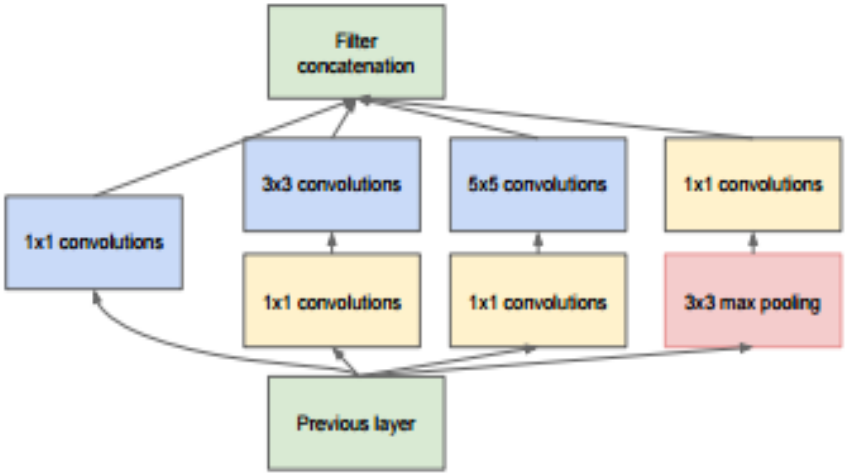


(a) Inception module, naïve version

Problem with the Naive inception architecture

- Higher up in the network, more correlated neurons are more spatially spread out.
- Thus, expect to use increasingly more 3X3 and 5X5 convolution maps to cover the optimal structure.
- But this is more expensive.
- Similar to Network-in-network, perhaps the use a layer of 1X1 convolutions for cross feature map-pooling will allow the capture of more complex interactions between the inputs (feature maps of the last layer), and thus, allow one to get away with lesser number of larger filters.

Inception 2.0



(b) Inception module with dimensionality reduction

GoogLeNet

- https://github.com/BVLC/caffe/issues/1106 - Implementation of inception layer in Caffe.
- Look at table 1 of paper for used layers.