DATA SCIENCE INTERVIEW PREPARATION (30 Days of Interview Preparation)

Day21

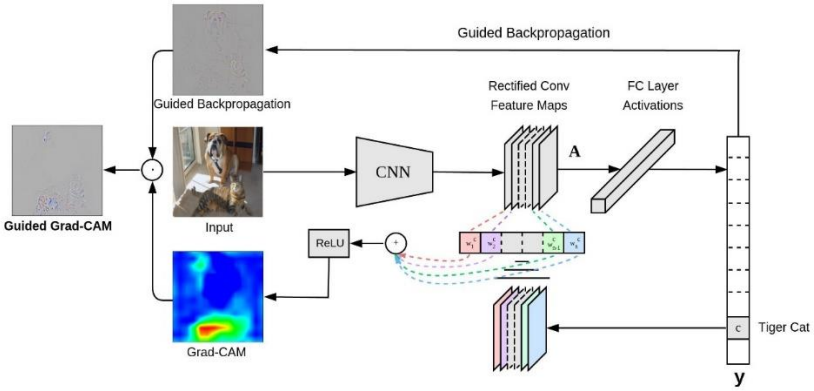
Q1. Explain Grad-CAM architecture?

Answer:

According to the research paper, "We propose a technique for making Convolutional Neural Network (CNN)-based models more transparent by visualizing input regions that are 'important' for predictions – producing *visual explanations*. Our approach is called Gradient-weighted Class Activation Mapping (Grad-CAM), which uses class-specific gradient information to localize the crucial regions. These localizations are combined with the existing pixel-space visualizations to create a new high-resolution, and class-discriminative display called the Guided Grad-CAM. These methods help better to understand CNN-based models, including image captioning and the apparent question answering (VQA) models. We evaluate our visual explanations by measuring the ability to discriminate between the classes and to inspire trust in humans, and their correlation with the occlusion maps. Grad-CAM provides a new way to understand the CNN-based models."

A technique for making CNN(Convolutional Neural Network)-based models more transparent by visualizing the regions of input that are "important" for predictions from these models — or visual explanations.

This visualization is both high-resolution (when the class of interest is 'tiger cat,' it identifies crucial 'tiger cat' features like stripes, pointy ears and eyes) and class-discriminative (it shows the 'tiger cat' but not the 'boxer (dog)').



Q2.Explain squeeze-net architecture?

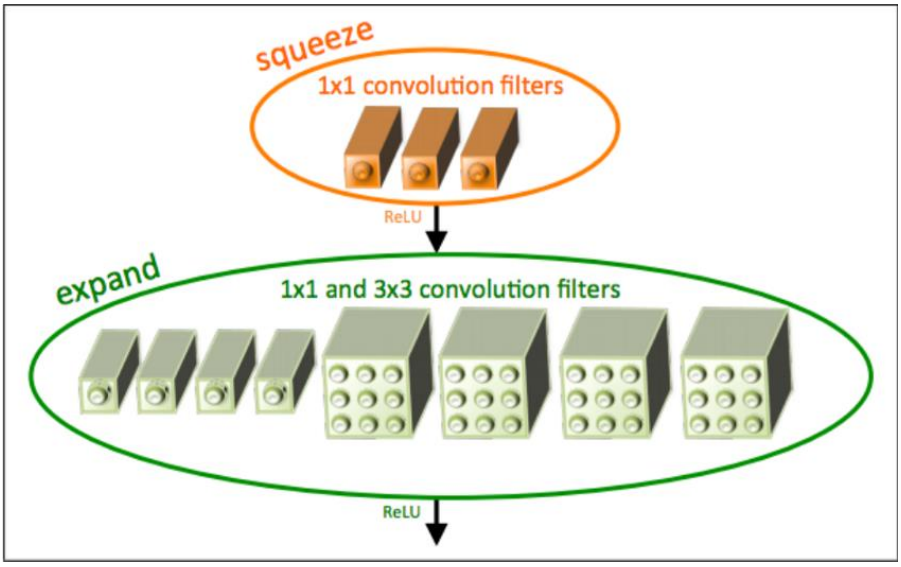
Answer:

Nowadays, technology is at its peak. Self-driving cars and IoT is going to be household talks in the next few years to come. Therefore, everything is controlled remotely, say, e.g., in self-driving cars, we will need our system to communicate with the servers regularly. So accordingly, if we have a model that has a small size, then we can quickly deploy it in the cloud. So that's why we needed an architecture that is less in size and also achieves the same level of accuracy that other architecture achieves.

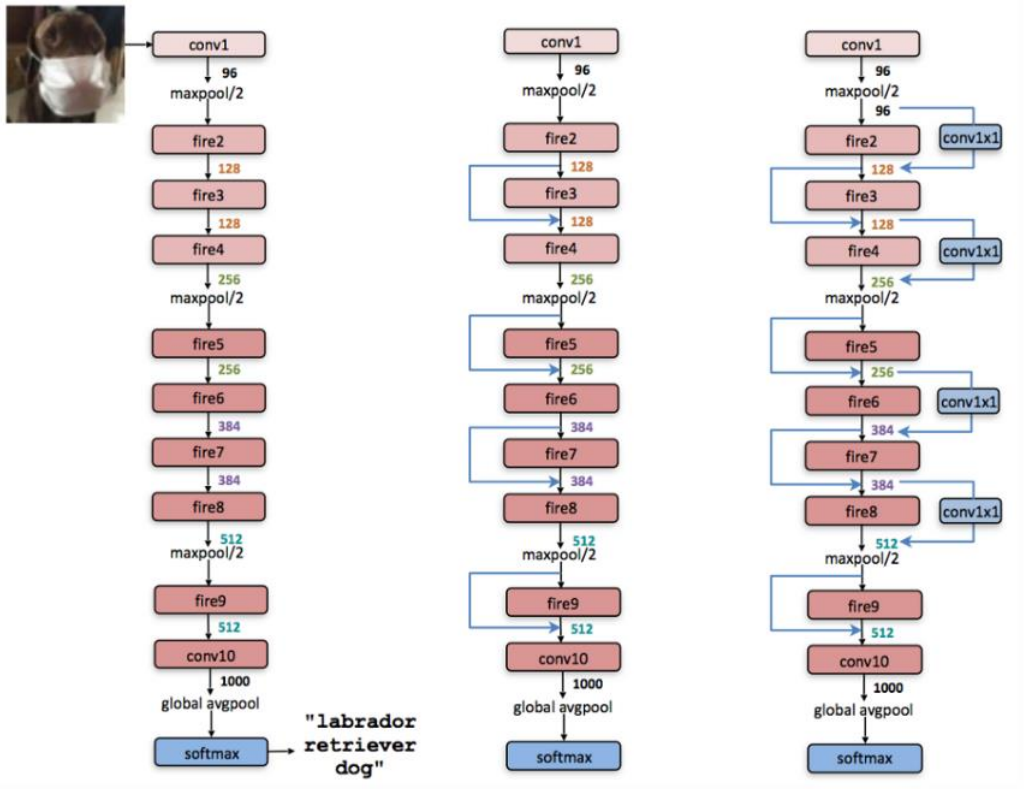
It's Architecture

- **Replace 3x3 filters with 1x1 filter-** We plan to use the maximum number of 1x1 filters as using a 1X1 filter rather than a 3X3 filter can reduce the number of parameters by 9X. We may think that replacing 3X3 filters with 1X1 filters may perform badly as it has less information to work on. But this is not a case. Typically 3X3 filter may capture the spatial information of pixels close to each other while the 1X1 filter zeros in on pixel and captures features amongst its channels.

- **Decrease number of input channels to 3x3 filters-** to maintain a small total number of parameters in a CNN, and it is crucial not only to decrease the number of 3x3 filters, but also to decrease the number of input channels to 3x3 filters. We reduce the number of input channels to 3x3 filters using *squeeze layers.* The author of this paper has used a term called the "*fire module,"* in which there is a squeeze layer and an expanded layer. In the squeeze layer, we are using 1X1 filters, while in the expanded layer, we are using a combo of 3X3 filters and 1X1 filters. The author is trying to limit the number of inputs to 3X3 filters to reduce the number of parameters in the layer.



- **Downsample late in a network so that convolution layers have a large activation map-** Having got an intuition about contracting the sheer number of parameters we are working with, how the model is getting most out of the remaining set of parameters. The author in this paper has downsampled the feature map in later layers, and this increases the accuracy. But this is an excellent contrast to networks like VGG where a large feature map is taken, and then it gets smaller as network approach towards the end. This different approach is too interesting, and they cite the paper by K. He and H. Sun that similarly applies delayed downsampling that leads to higher classification accuracy.
  This architecture consists of the fire module, which enables it to bring down the number of parameters.



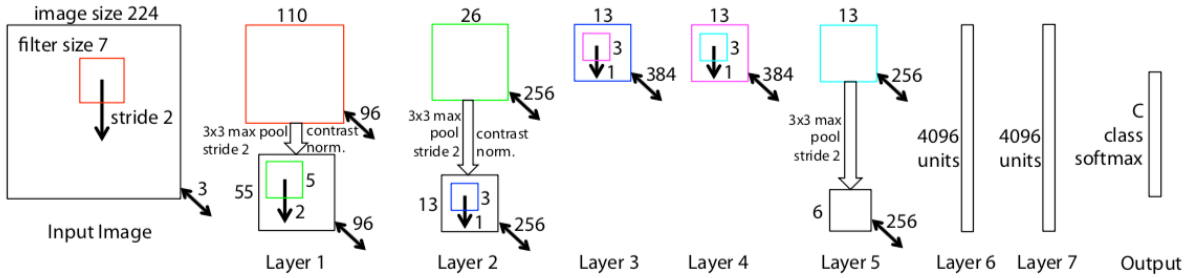
And other thing that surprises me is the lack of fully connected layers or dense layers at the end, which one will see in a typical CNN architecture. The dense layers, in the end, learn all the relationships between the high-level features and the classes it is trying to identify. The fully connected layers are designed to learn that noses and ears make up a face, and wheels and lights indicate cars. However, in this architecture, that extra learning step seems to be embedded within the transformations between various “fire modules.”

CNN architecture	Compression Approach	Data Type	Original → Compressed Model Size	Reduction in Model Size vs. AlexNet	Top-1 ImageNet Accuracy	Top-5 ImageNet Accuracy
AlexNet	None (baseline)	32 bit	240MB	1x	57.2%	80.3%
AlexNet	SVD [5]	32 bit	240MB → 48MB	5x	56.0%	79.4%
AlexNet	Network Pruning [11]	32 bit	240MB → 27MB	9x	57.2%	80.3%
AlexNet	Deep Compression [10]	5-8 bit	240MB → 6.9MB	35x	57.2%	80.3%
SqueezeNet (ours)	None	32 bit	4.8MB	**50x**	57.5%	80.3%
SqueezeNet (ours)	Deep Compression	8 bit	4.8MB → 0.66MB	**363x**	57.5%	80.3%
SqueezeNet (ours)	Deep Compression	6 bit	4.8MB → 0.47MB	**510x**	57.5%	80.3%

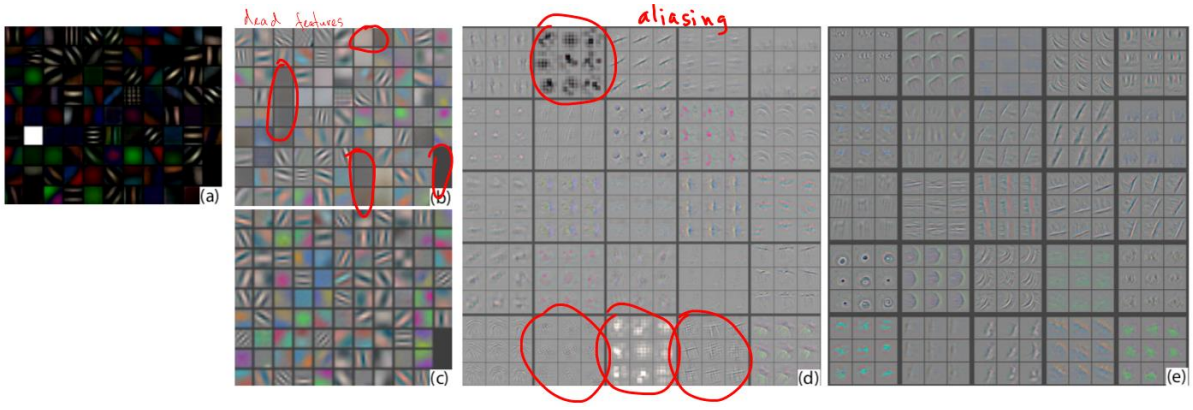
The squeeze-net can accomplish an accuracy nearly equal to AlexNet with 50X less number of parameters. The most impressive part is that if we apply Deep compression to the already smaller model, then it can reduce the size of the squeeze-net model to 510x times that of AlexNet.

Q3.ZFNet architecture

Answer:



The architecture of the network is an optimized version of the last year's winner - AlexNet. The authors spent some time to find out the bottlenecks of AlexNet and removing them, achieving superior performance.



(a): First layer ZFNET features without feature scale clipping. (b): the First layer features from AlexNet. Note that there are lot of dead features - ones where the network did not learn any patterns. (c): the First layer features for ZFNet. Note that there are only a few dead features. (d): Second layer features from AlexNet. The grid-like patterns are so-called aliasing artifacts. They appear when

receptive fields of convolutional neurons overlap, and neighboring neurons learn similar structures. (e): 2nd layer features for ZFNet. Note that there are no aliasing artifacts. Source: original paper.

In particular, they reduced the filter size in the 1st convolutional layer from 11x11 to 7x7, which resulted in fewer dead features learned in the first layer (see the image below for an example of that). A dead feature is a situation where a convolutional kernel fails to learn any significant representation. Visually it looks like a monotonic single-color image, where all the values are close to each other.

In addition to changing the filter size, the authors of FZNet have doubled the number of filters in all convolutional layers and the number of neurons in the fully connected layers as compared to the AlexNet. In the AlexNet, there were 48-128-192-192-128-2048-2048 kernels/neurons, and in the ZFNet, all these doubled to 96-256-384-384-256-4096-4096. This modification allowed the network to increase the complexity of internal representations and as a result, decrease the error rate from 15.4% for last year's winner, to 14.8% to become the winner in 2013.

Q4. What is NAS (Neural Architecture Search)?

Answer:

Developing the neural network models often requires significant architecture engineering. We can sometimes get by with transfer learning, but if we want the best possible performance, it's usually best to design your network. This requires specialized skills and is challenging in general; we may not even know the limits of the current state-of-the-art(SOTA) techniques. Its a lot of trial and error, and experimentation itself is time-consuming and expensive.

This is the NAS(Neural Architecture Search) comes in. NAS(Neural Architecture Search) is an algorithm that searches for the best neural network architecture. Most of the algorithms work in the following way. Start off by defining the set of "building blocks" that can be used for our network. E.g., the state-of-the-art(SOTA) NASNet paper proposes these commonly used blocks for an image recognition network-

- identity
- 1x7 then 7x1 convolution
- 3x3 average pooling
- 5x5 max pooling
- 1x1 convolution
- 3x3 depthwise-separable conv
- 7x7 depthwise-separable conv
- 1x3 then 3x1 convolution
- 3x3 dilated convolution
- 3x3 max pooling
- 7x7 max pooling
- 3x3 convolution
- 5x5 depthwise-seperable conv

In the NAS algorithm, the controller Recurrent Neural Network (RNN) samples the building blocks, putting them together to create some end to end architecture. Architecture generally combines the same style as state-of-the-art(SOTA) networks, such as DenseNets or ResNets, but uses a much different combination and the configuration of blocks.

This new network architecture is then trained to convergence to obtain the least accuracy on the held-out validation set. The resulting efficiencies are used to update the controller so that the controller will generate better architectures over time, perhaps by selecting better blocks or making better connections. The controller weights are updated with a policy gradient. The whole end-to-end setup is shown below.

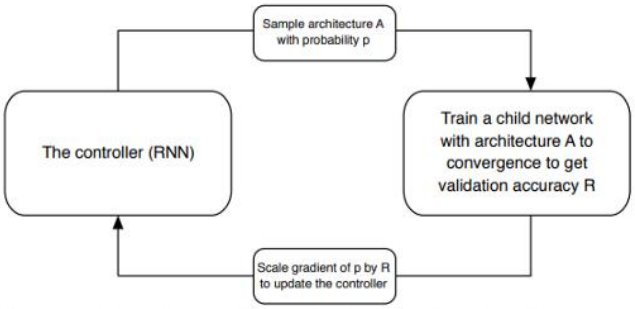
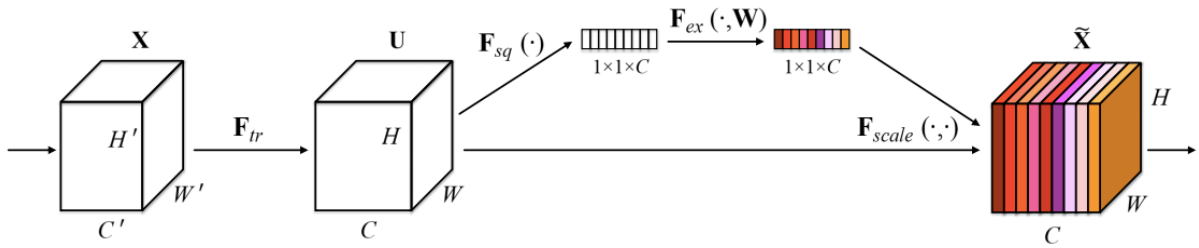


Figure 1. Overview of Neural Architecture Search [71]. A controller RNN predicts architecture A from a search space with probability p. A child network with architecture A is trained to convergence achieving accuracy R. Scale the gradients of p by R to update the RNN controller.

It's a reasonably intuitive approach! In simple means: have an algorithm grab different blocks and put those blocks together to make the network. Train and test out that network. Based on our results, adjust the blocks we used to make the network and how you put them together!

Q5. What is SENets?

Answer:



SENets stands for Squeeze-and-Excitation Networks introduces a building block for CNNs that improves channel interdependencies at almost no computational cost. They have used in the 2017 ImageNet competition and helped to improve the result from last year by 25%. Besides this large performance boost, they can be easily added to existing architectures. The idea is this:

Let's add parameters to each channel of the convolutional block so that the network can adaptively adjust the weighting of each feature map.

As simple as may it sound, this is it. So, let's take a closer look at why this works so well.

Why it works too well?

CNN's uses its convolutional filters to extract hierarchal information from the images. Lower layers find little pieces of context like high frequencies or edges, while upper layers can detect faces, text, or other complex geometrical shapes. They extract whatever is necessary to solve the task precisely.

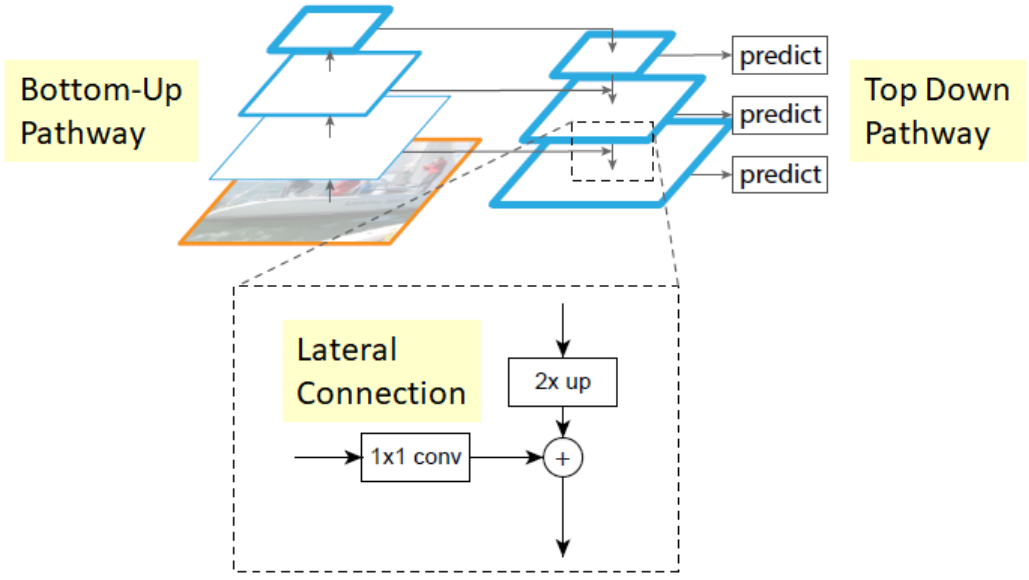
All of this works by fusing spatial and channel information of an image. The different filters will first find the spatial features in each input channel before adding the information across all available output channels.

All we need to understand for now is that the network weights each of its channels equally when creating output feature maps. It is all about changing this by adding a content-aware mechanism to weight each channel adaptively. In its too basic form, this could mean adding a single parameter to each channel and giving it linear scalar how relevant each one is.

However, the authors push it a little further. First, they get the global understanding of each channel by squeezing feature maps to a single numeric value. This results in the vector of size *n*, where *n* is equal to the number of convolutional channels. Afterward, it is fed through a two-layer neural network, which outputs a vector of the same size. These *n* values can now be used as weights on the original features maps, scaling each channel based on its importance.

Q6. Feature Pyramid Network (FPN)

Answer:



The Bottom-Up Pathway

The bottom-up pathway is feedforward computation of backbone ConvNet. It is known as one pyramid level is for each stage. The output of last layer of each step will be used as the reference set of feature maps for enriching the top-down pathway by lateral connection.

Top-Down Pathway and Lateral Connection

- The higher resolution features are upsampled spatially coarser, but semantically stronger, feature maps from higher pyramid levels. More particularly, the spatial resolution is upsampled by a factor of 2 using nearest neighbor for simplicity.
- Each lateral connection adds feature maps of the same spatial size from the bottom-up pathway and top-down pathway.
- Specifically, **the feature maps from the bottom-up pathway undergo 1×1 convolutions to reduce channel dimensions.**
- And **feature maps from the bottom-up pathway and top-down pathway are merged by element-wise addition.**

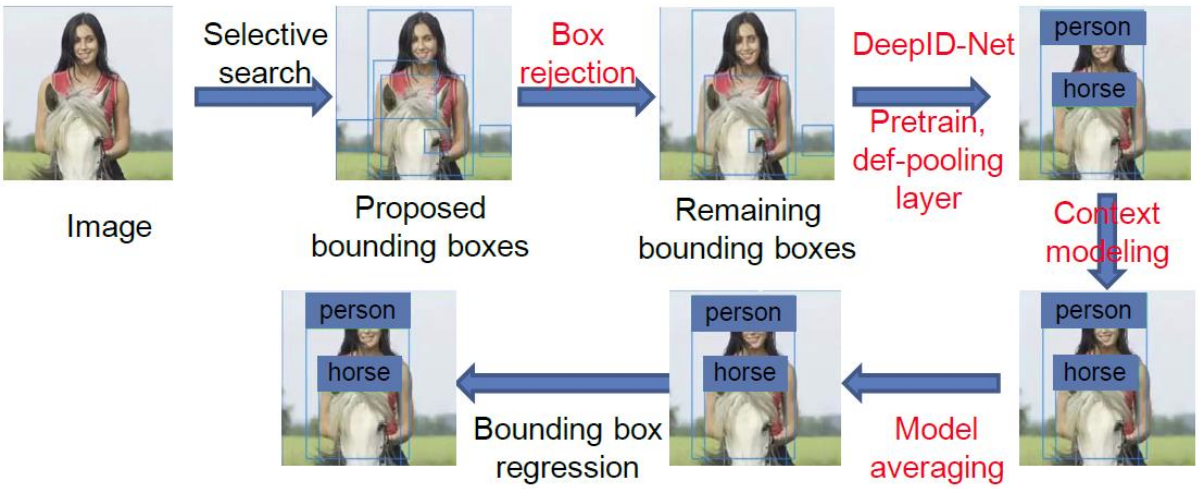
Prediction in FPN

- Finally, **the 3×3 convolution is appended on each merged map to generate a final feature map, which is to reduce the aliasing effect of upsampling.** This last set of feature maps is called {P2, P3, P4, P5}, corresponding to {C2, C3, C4, C5} that are respectively of same spatial sizes.
- Because all levels of pyramid use shared classifiers/regressors as in a traditional featured image pyramid, feature dimension at output d is fixed with $d = 256$. Thus, all extra convolutional layers have 256 channel outputs.

Q7. DeepID-Net(Def-Pooling Layer)

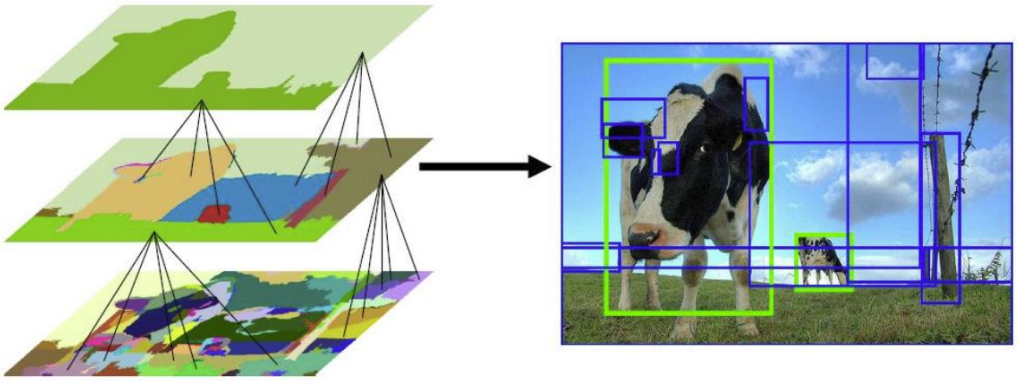
Answer:

A new def-pooling (deformable constrained pooling) layer is used to model the deformation of the object parts with geometric constraints and penalties. That means, except detecting the whole object directly, it is also important to identify object parts, which can then assist in detecting the whole object.



The steps in **black** color are the **old stuff** that **existed in R-CNN**. The stages in **red** color do not appear **in R-CNN**.

1. **Selective Search**

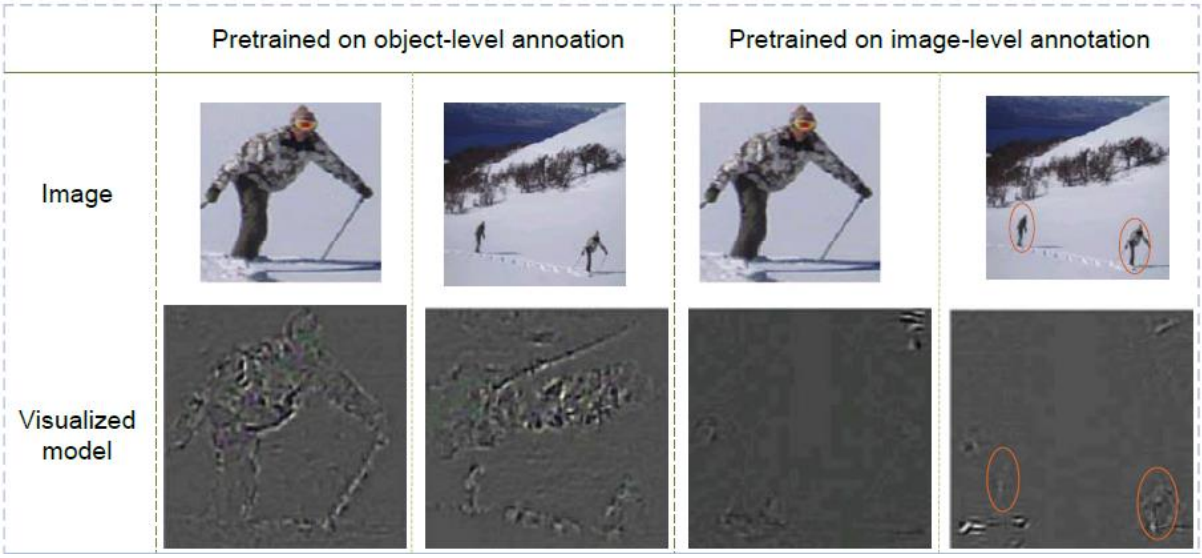


- First, color similarities, texture similarities, regions size, and region filling are used as **non-object-based segmentation**. Therefore you obtain **many small segmented areas** as shown at the bottom left of the image above.
- Then, the bottom-up approach is used that **small segmented areas are merged to form the larger segment areas.**
- Thus, **about 2K regions, proposals (bounding box candidates) are generated,** as shown in the above image.

2. Box Rejection

R-CNN is used to reject bounding boxes that are most likely to be the background.

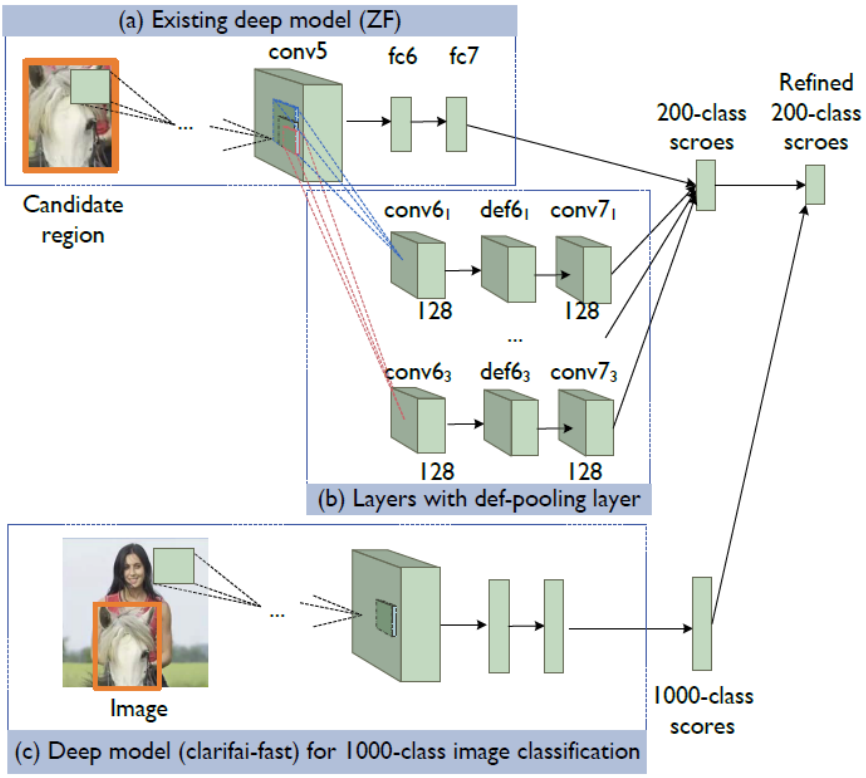
3. Pre train Using Object-Level Annotations

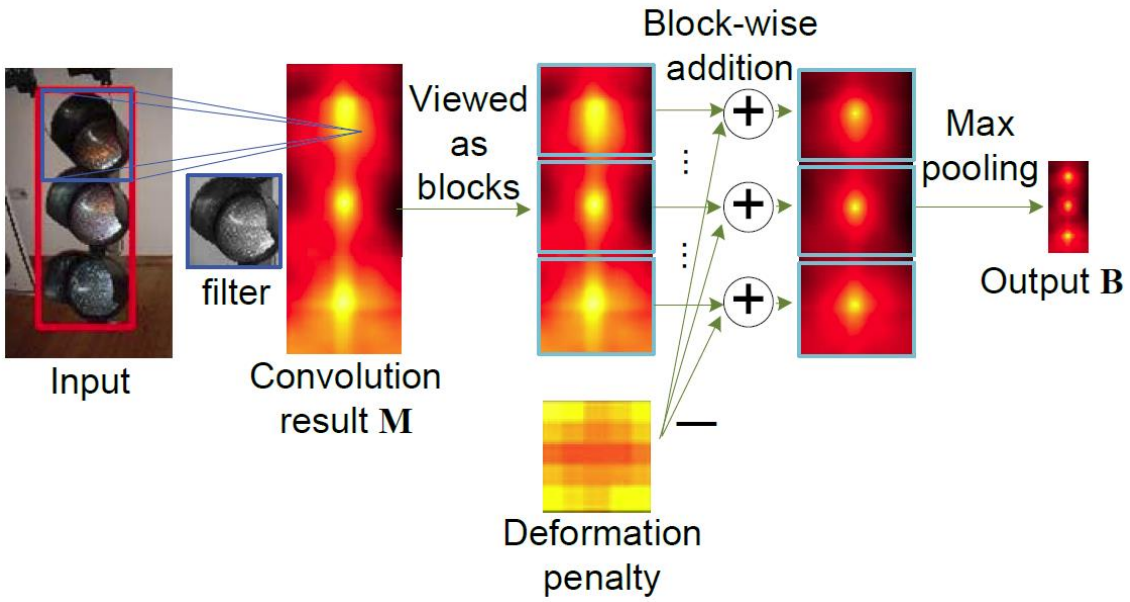


Usually, pretraining is on **image-level annotation**. It is **not good when an object is too small within the image** because the object should occupy a large area within the bounding box created by the selective search.

Thus, **pretraining is on object-level annotation**. And **the deep learning(DL) model can be any models** such as ZFNet, VGGNet, and GoogLeNet.

4. Def-Pooling Layer





$$b_c^{(x,y)} = \max_{\delta_x,\delta_y \in \{-R,\cdots,R\}} \{m_c^{\mathbf{z}_{\delta_x,\delta_y}} - \sum_{n=1}^{N} a_{c,n} d_{c,n}^{\delta_x,\delta_y}\}$$

$$\text{where } \mathbf{z}_{\delta_x,\delta_y} = (s_x \cdot x + \delta_x, s_y \cdot y + \delta_y).$$

For the def-pooling path, output from conv5, goes through the Conv layer, then goes through the def-pooling layer, and then has a max-pooling layer.

In simple terms, the summation of ac multiplied by dc,n, is the 5×5 deformation penalty in the figure above. The penalty of placing object part from assumed the central position.

By training the DeepID-Net, object parts of the object to be detected will give a high activation value after the def-pooling layer if they are closed to their anchor places. And this output will connect to 200-class scores for improvement.

5. Context Modeling

In object detection tasks in ILSVRC, there are 200 classes. And there is also the classification competition task in ILSVRC for classifying and localizing 1000-class objects. The contents are more diverse compared with the object detection task. Hence, **1000-class scores, obtained by classification network, are used to refine 200-class scores.**

6. The Model Averaging-

Multiple models are used to increase the accuracy, and **the results from all models are averaged**. This technique has been used since AlexNet, LeNet, and so on.

7. Bounding Box Regression

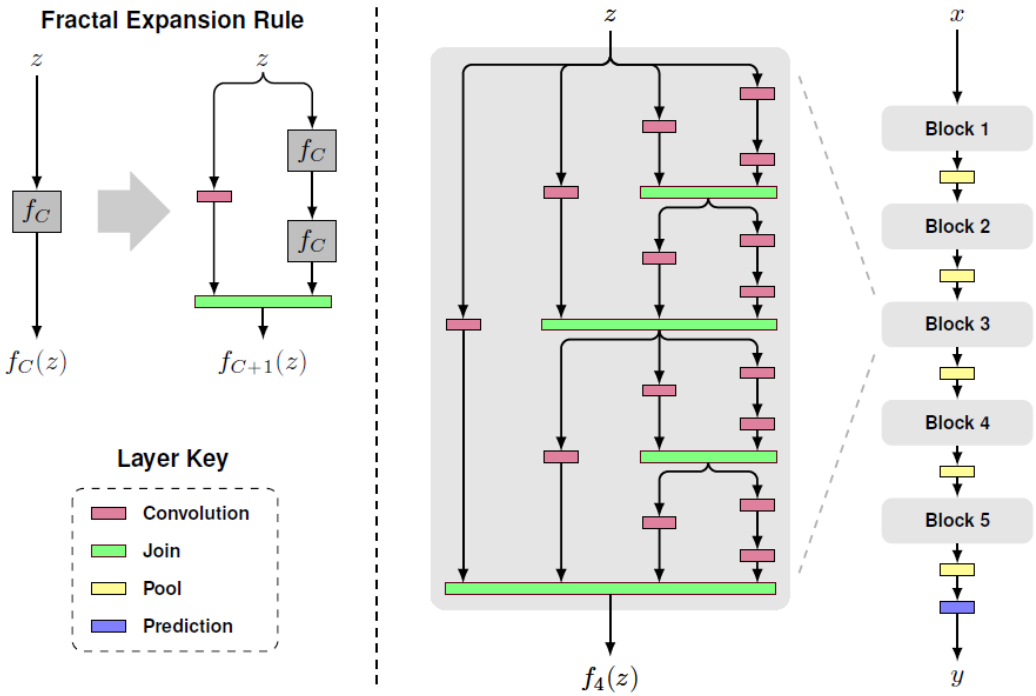
Bounding box regression is to **fine-tune the bounding box location**, which has been used in R-CNN.

Q8. What is FractalNet Architecture?

Answer:

In2015, after the invention of ResNet, with numerous champion won, there are plenty of researchers working on how to improve the ResNet, such as Pre-Activation ResNet, RiR, RoR, Stochastic Depth, and WRN. In this story, conversely, a non-residual-network approach, FractalNet, is shortly reviewed. When VGGNet is starting to degrade when it goes from 16 layers (VGG-16) to 19 layers (VGG-19), FractalNet can go up to 40 layers or even 80 layers.

Architecture



In the above picture: A Simple Fractal Expansion (on Left), Recursively Stacking of Fractal Expansion as One Block (in the Middle), 5 Blocks Cascaded as FractalNet (on the Right)

For the base case, f1(z) is the convolutional layer:

$$f_1(z) = \text{conv}(z)$$

After that, recursive fractals are:

$$f_{C+1}(z) = [(f_C \circ f_C)(z)] \oplus [\text{conv}(z)]$$

Where C is a number of columns as in the middle of the above figure. The number of the convolutional layers at the deepest path within the block will have 2^(C-1). In this case, **C=4**, thereby, a number of convolutional layers are 2^3=8 layers.

For the **join layer** (green), the **element-wise mean** is computed. It is not concatenation or addition.

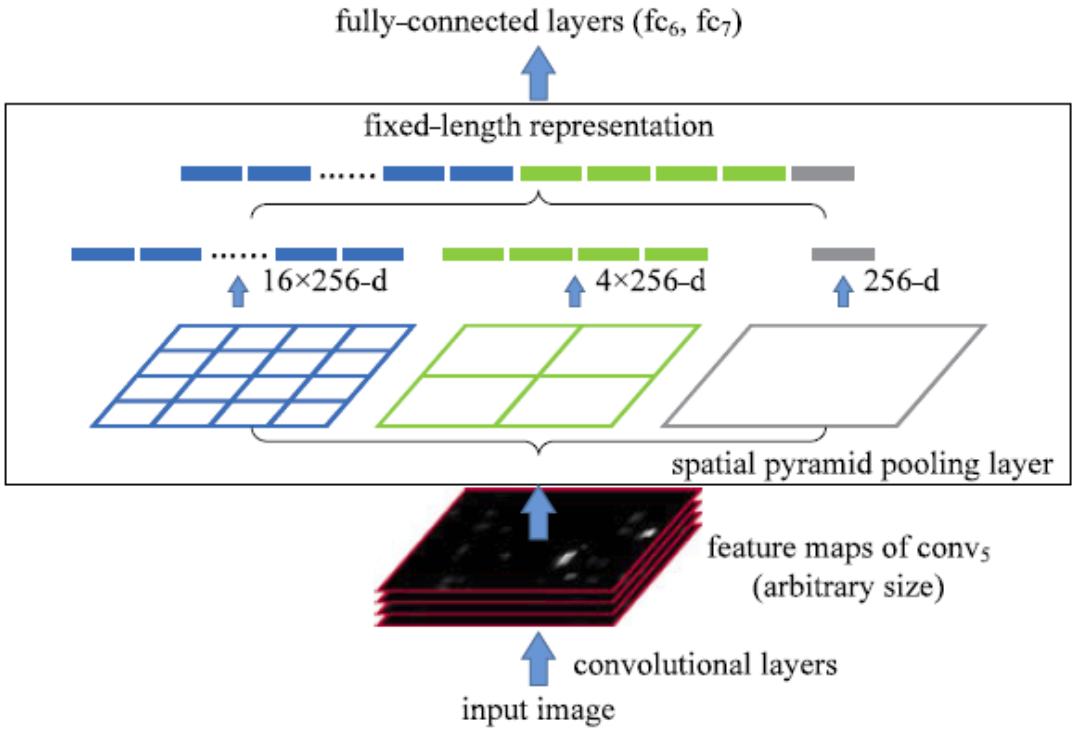
With five blocks (**B=5**) cascaded as FractalNet at the right of the figure, then the number of convolutional layers at the most profound path within the whole network is B×2^(C-1), i.e., 5×2^3=**40 layers**.

In between 2 blocks, 2×2 max pooling is done to reduce the size of feature maps. Batch Norm and ReLU are used after each convolution.

Q9. What is the SSPNet architecture?

Answer:

SPPNet has introduced the new technique in CNN called **Spatial Pyramid Pooling (SPP)** at the transition of the convolutional layer and fully connected layer. This is a work from **Microsoft**.



Conventionally, at the transformation of the Conv layer and FC layer, there is one single pooling layer or even no pooling layer. In SPPNet, it suggests having **multiple pooling layers with different scales**.

In the figure, **3-level SPP** is used. Suppose conv5 layer has 256 feature maps. Then at the SPP layer,

1. first, each feature map is **pooled to become one value (which is grey)**. Thus **256-d vector is formed**.
2. Then, each feature map is **pooled to have four values (which is green)**, and form the **4×256-d vector**.
3. Similarly, each feature map is **pooled to have 16 values (in blue)**, and form the **16×256-d vector**.
4. The **above three vectors are concatenated to form a 1-d vector**.
5. Finally, this **1-d vector is going into FC layers** as usual.

With SPP, you don't need to crop the image to a fixed size, like AlexNet, before going into CNN. **Any image sizes can be inputted.**