Mordern History of Object Recognition Infographics

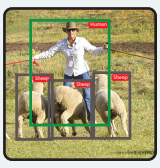
Acronyms

Image Classification
Classify an image based on the dominant object inside it.

datasets: MNIST, CIFAR, ImageNet

Object Localization
Localize the image region that contains the dominant object

datasets: ImageNet



Object Recognition
Localize and classify all objects appearing in an image

datasets: PASCAL, COCO

Semantic Segmentation
Label each pixel of an image by the object class that it belongs to, such as human, sheep, and grass in the example.

datasets: PASCAL, COCO

Instance Segmentation
Label each pixel of an image by the object class and object instance that it belongs to. Background pixels are usually ignored.

datasets: PASCAL, COCO

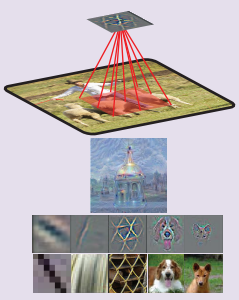
Keypoint Detection
Detect locations of a set of predefined keypoints of an object, such as keypoints in human body, or human face.

datasets: COCO

1955 1980 1990 2010 2012

Smith Huber
Yan Lecun
Bengjo
Alex Graves
Hinton
Smith Huber
Yan Lecun
Bengjo
Alex Graves
Hinton

Important CNN Concepts



Feature *(pattern, activation of a neuron, feature detector)* [3] [4]

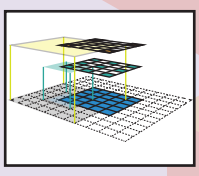
A hidden neuron that is activated when a particular pattern (feature) is presented in its input region (receptive field).

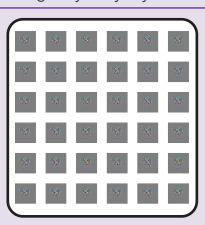
The pattern that a neuron is detecting can be visualized by (1) optimizing its input region to activate the neuron the most (deep dream), (2) visualizing the gradient or guided gradient of the neuron activation on its input pixels (back probagation and guided back probagation), (3) visualizing a set of image regions in the training data that activate the neuron the most

Receptive Field *(input region of a feature)* [1] [2]

The region of the input image that affects the activation of a feature. In other words, it is the region that the feature is looking at.

Generally, a feature in a higher layer has a bigger receptive field, which allows it to learn to capture a more complex/abstract pattern. The ConvNet architecture determines how the receptive field change layer by layer.



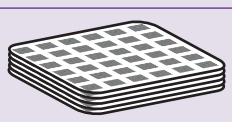


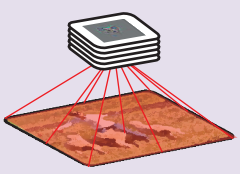
Feature Map *(a channel of a hidden layer)* [1]

A set of features that created by applying the same feature detector at different location of an input map in a sliding window fashion (i.e. convolution). Features in the same feature map have the same receptive size and look for the same pattern, but at different locations. This creates the spatial invariance properties of a CNN

Feature Volume *(a hidden layer in CNNs)*

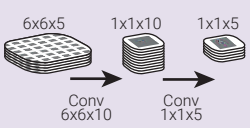
A set of feature maps, each map searches for a particular feature at a fixed set of locations on the input map. All features have the same receptive field size.





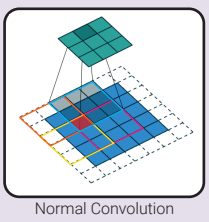
Fully connected layer as Feature Volume [5]

Fully connected layers (usually attached to the end of a ConvNet for classification) with k hidden nodes can be seen as a *1x1xk* feature volume. This feature volume has one feature in each feature map, and its receptive field covers the whole image. The weight matrix W in an FC layer can be converted to an CNN filter. Convoluting the filter kernel *wxhxk* to a CNN feature volume wxhxd creates a *1x1xk* feature volume (k-node FC layer). Convoluting a *1x1xk* filter kernel to a *1x1xd* feature volume creates a *1x1xk* feature volume. Replacing fully connected layers by convolution layers allows us to apply a ConvNet to an image with arbitrary size.



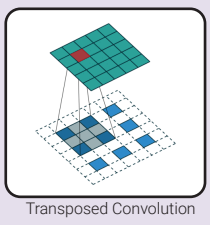
Transposed Convolution *(fractional strided convolution, deconvolution, upsampling)* [2]

The operation that back-probagates the gradient of a convolution operation. In other words, it is the backward pass of a convolution layer. A transposed convolution can be implemented as a normal convolution with zero inserted between the input features. A convolution with filter size k, stride s and zero padding p has an associated transposed convolution with filter size k'=k, stride s'=1, zero padding p'=k-p-1, and s-1 zeros inserted between each input unit.



Normal Convolution
k = 3, s=2, p = 1

On the left, the red input unit contributes to the activation of the 4 top left output units (4 colored squares), therefore it receives gradient from these output units. This gradient backpropagation can be implemented by the transposed convolution shown on the right.

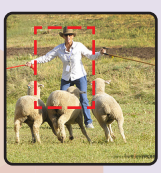


Transposed Convolution
k' = 3, s'=1, p = 1, insert=1

End-To-End object recognition pipeline *(end-to-end learning/system)*

A object recognition pipeline that all stages (pre-processing, region proposal generation, proposal classification, post-processing) can be trained altogether by optimizing a ***single objective function***, which is a differentiable function of all stages' variables. This is the opposite of the traditional object recognition pipeline, which connects stages in a non-differentiable fashion. In these systems, we do not know how changing a stage's variable can affect the overall performance, so that each stage must be trained (or heuristically programmed) independently from each other.

Important Object Recognition Concepts

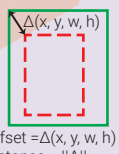


Bounding box proposal *(region of interest, region proposal, box proposal)*

A rectangular region of the input image that potentially contains an object inside. These proposals can be generated by some heuristics search: objectness, selective search, or by a region proposal network

A bounding box can be represented as a 4-element vector, either storing its two corner coordinates (x0, y0, x1, y1), or (commonly) storing its center location and its width and height (x, y, w, h). A bounding box is also usually accompanied by a confidence score of how likely the box contains an object, which NMS requires.

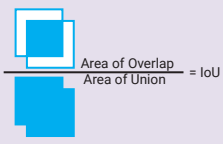
The difference between two bounding boxes is usually measured by the L2 distance of their vector representations. w and h can be log-transformed before the distance calculation.



Offset =Δ(x, y, w, h)
Distance = ||Δ||

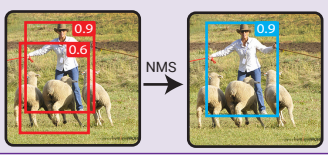
Intersection over Union *(IoU, Jaccard similarity):*

A metric that measures the similarity between two bounding boxes = their overlapping area over their union area.



Non Maxium Suppression *(NMS)*

A common algorithm to merge overlapping bounding boxes (proposals or detections). Any bounding box that significantly overlap (IoU > IoU_threshold) with a higher-confident bounding box is suppressed (removed).



Bounding box regression *(bounding box refinement)*

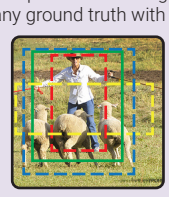
By looking at an input region, we can infer the bounding box that better fit the object inside, even if the object is only partly visible. The example on the right illustrates the possibility of inferring the ground truth box only by looking at part of an object. Therefore, one regressor can be trained to predict the offset Δ(x, y, w, h) between the input bounding box (input region) and the ground truth box. if we have one regressor for each object class, it is called class-specific regression, otherwise (one regressor for all) it is class-agnostic. A bounding box regressor is often accompanied by a ***bounding box classifier*** to estimate the confidence of object existence in the predicted box. The classifier can also be class-specific or class-agnostic. Without prior box, the input region here plays the role of a prior box.

Prior box *(default box, anchor box)*

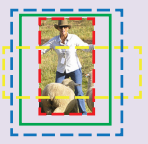
Instead of using the input region as the only prior box, we can train multiple bounding box regressors, each has a different prior box and learns to predict the offset between its own prior box and the ground truth box. By looking at the same input region, regressors with different prior boxes can learn to predict bounding boxes with different properties (aspect ratio, scale, locations). Prior boxes can be predefined relatively to the input region, or learned by clustering. An appropriate box matching strategy is crucial to make the training converge.

Box Matching Strategy

We cannot expect a bounding box regressor to be able to predict a bounding box of an object that is too far away from its input region and/or its prior box. Therefore, we need a box matching strategy to decide which prior box is matched with a ground truth box. Each match is a training example to regress. Possible strategies: (Multibox) matching each ground truth box with one prior box with highest IoU; (SSD, FasterRCNN) matching a prior box with any ground truth with IoU higher than 0.5.



One region proposal with 3 prior boxes and one ground truth box



The 3 bounding box regressors only see the input region and try to infer the ground truth box from their prior boxes

In Multibox strategy, the ground truth box is matched with the prior box with highest IoU

Hard negative example mining

For each prior box, there is a bounding box classifier that estimates the likelihood of having an object inside. After box matching, all matched prior boxes are positive examples for the classifier. All other prior boxes are negatives. We cannot use all the hard negative examples, since there is a significant imbalance between the positives and negatives. Possible solutions: pick randomly negative examples (FasterRCNN), or pick the ones that the classifier makes the most serious error (SSD), so that the ratio between the negatives and positives is at roughly 3:1.

References

1. *A guide to convolution arithmetic for deep learning. arXiv*
2. *A guide to receptive field arithmetic for CNN. medium*
3. *Visualizing and Understanding Convolutional Networks. arXiv*
4. *Inceptionism: Going Deeper into Neural Networks.*
5. *CS231n Convolutional Neural Networks for Visual Recognition*
6.
7.
8.

Author Info:
Original Link: