Notes - Fully Convolutional Network (FCN)

Wednesday, December 12, 2018 10:32 PM

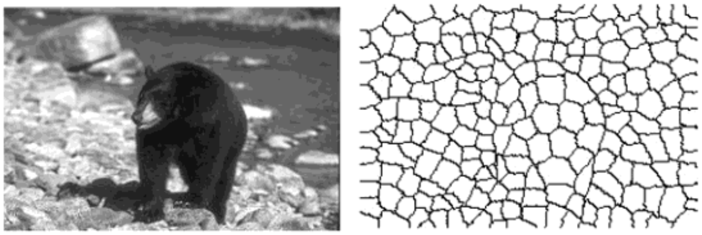
What is FCN?
It is a machine learning algorithm used for the computer vision application especially semantic segmentation.

What is the purpose?
The purpose of FCN is to segment the images and assign pixels belonging to an object to a specific class label.

What is the advantage of FCN?
When compared to the approaches that were proposed prior to FCN, FCN exceeds the performance of the state of the art algorithm and can be trained end to end.

What is the common pipeline of previous approaches?
Previous approaches have used pre-processing and post-processing. Pre-processing includes superpixels, proposals and post-processing includes refinement by random-fields or local classifiers.

What is superpixel? (One of the techniques used by previous approach)
Superpixel is a perceptually meaningful entity that can be obtained from a low-level grouping process. Normalized cuts is a classical region segmentation algorithm which used spectral clustering to exploit pairwise brightness, color and texture affinities between pixels. This algorithm can be applied to oversegment images to obtain superpixels. Following example shows such a superpixel map with the number of clusters 200.


Original Image Superpixel map with k=200

How Fully Convolutional network uses the CNN architecture for semantic segmentation?
Deep feature hierarchies created by Convolutional Neural Network encode location and semantics in a nonlinear local-to-global pyramid i.e. the deep layers (feature layers near the end) in the CNN can produce the coarse semantic information and the shallow layers (initial feature layers) provide the fine, appearance related information. These two layers are combined using the skip architecture.

What are the previous approaches related to FCN?
The approach used by FCN combines the successes of deep nets for image classification and transfer learning.
<u>Fully convolution approach:</u>
This approach is initially used for the extension of convnets to arbitrary sized inputs, where the classic LeNet is used to recognize the string of digits. Here the input is considered as one dimensional strings and Viterbi decoding is used to obtain the results. The approach is also used in a lot of coarse multi-class segmentation in medical applications, sliding window detection, image restoration etc.,
<u>Dense predictions with convnets:</u>
Several approaches have used the convnets to do the semantic segmentation. Common elements used by them include the following:

1. Having small models and restricting the capacity and receptive fields
2. Patch wise training
3. Post-processing using the super-pixel projection, random field regularization, filtering or local classification
4. Input shifting and output interlacing for dense output
5. Multi-scale pyramid processing
6. Saturating tanh nonlinearities
7. Ensembles

But the FCN does not use any of these machineries. FCN uses patch wise training and shift-and-stitch dense output, in-network up-sampling.

FCN uses fusion architecture that fuses the features across layers to define a nonlinear local-to-global representation which is then tuned end to end.

How does the convolution layer can be described mathematically?

The basic components of convnets are convolution, pooling and activation functions. They operate on local input regions and depend only on relative spatial coordinates.

x_{ij} for the data vector at location (i; j) in a particular layer, and y_{ij} for the following layer can be written using the following expression

$$\mathbf{y}_{ij} = f_{ks}\left(\{\mathbf{x}_{si+\delta i, sj+\delta j}\}_{0 \leq \delta i, \delta j \leq k}\right)$$

Where k is the kernel size, s is the stride or subsampling factor and f_{ks} determine the layer type

- Matrix multiplication for convolution or average pooling
- Spatial max for max pooling
- Elementwise nonlinearity for an activation function.

How the FCN differs from a common deep network?

A general deep net computes a general nonlinear function, but FCN with only layers of convolutional form computes a nonlinear filter. So an FCN operates on an input of any size and produces an output of corresponding spatial dimensions.

How does the loss function and corresponding gradient looks in the FCN?

Loss function is a sum over the spatial dimension of the final layer,

$\ell(x; \theta) = \sum_{ij} \ell`(x_{ij}; \theta)$ , then the gradient will be the sum over the gradients of each of its spatial components. Then the stochastic gradient descent on $\ell$ computed on whole images will be the same as stochastic gradient descent on $\ell`$, taking all of the final layer receptive fields as a minibatch.

How to adapt classifiers for the dense prediction?

Recognition nets like LeNet, AlexNet and its successors take fixed size inputs and produce non-spatial outputs. Fully connected layers of the above models have fixed dimensions and throw away spatial coordinates. These fully connected layers can be viewed as convolutions with kernels that cover their entire input regions which then converts them into fully convolutional networks and output classification maps.

The spatial output maps of this convolutional models make them a natural choice for dense problems like semantic segmentation. Ground truth is available at every output cell, so, both the forward and backward passes are straight forward.

While converting these fully connected layers to fully convolution ones, the output dimensions are reduced by subsampling. This coarsens the output by reducing it from the size of the input by a factor equal to the pixel stride of the receptive fields of the output units.

References:

Fully Convolutional Network
Link: https://people.eecs.berkeley.edu/~jonlong/long_shelhamer_fcn.pdf

Superpixel
Link: http://ttic.uchicago.edu/~xren/research/superpixel/