Semantic Image Segmentation with Deep Convolutional Nets and Fully Connected CRFs

1 Citation

Chen, Liang-Chieh, et al. "Deeplab: Semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected crfs." IEEE transactions on pattern analysis and machine intelligence 40.4 (2018): 834-848.

https://arxiv.org/pdf/1412.7062.pdf

2 Abstract

We combine a CNN and Conditional Random Field (CRF) to set state of the art on PASCAL VOC 2012. Using the "hole" algorithm from the wavelet community lets us process 8 frames a second on a GPU.

3 Introduction

There are two reasons you cannot use a CNN directly on image segmentation. First, the pooling with stride operation downsamples the image and loses information. We use the "atrous" (with holes) algorithm to apply the CNN densely and efficiently. Second, a CNN isn't invariant to many spatial transformations. To mitigate this, we use a conditional random field (CRF) to model local interactions between pixels.

Our DeepLab system is great because it is fast (8 frames per second vs. next best option of 0.5 frames per second), it sets state of the art on PASCAL VOC 2012, and it's pretty simple (just CNN and CRF).

4 Related Work

Unlike previous methods, we operate directly on raw pixels rather than committing to a single segmentation.

Some works do segmentation purely in a CNN by upsampling back to the original image size.

We are also unique because our CRF operates directly on a pixel and uses long range dependencies.

5 Convolutional Neural Networks for Dense Image Labeling

Here's how we use the VGG-16 CNN as our feature extractor. We first convert the fully connected layers into convolutional layers so we can apply it densely. However, this yields a 32 pixel stride, and we'd like to get an 8 pixel stride. To achieve this, we (1) remove downsampling after last two max-pooling layers, and (2) add zeros to increase filter length 2x in convolutional layers and 4x convolutional layers - the ”hole algorithm” lets us do this efficiently.

We start with the pretrained (on ImageNet) weights, replace the 1000-way classifier with a 21-way classifier and optimize with sum of cross entropy loss over output pixels (we have to subsample ground truth images by 8 to match prediction size). At test-time, we use bilinear interpolation to scale up the predicted image back to original image size.

Thanks to the ”hole algorithm”, we take just 10 hours to train on a GPU instead of days.

The first fully-connected layer (when converted to a convolutional layer) becomes a performance bottleneck because it has 4096 7×7 kernels. We subsample this to reduce computational load and we also use 1024 hidden units instead of 4096.

6 Detailed Boundary Recovery: Fully-Connected Conditional Random Fields and Multi-Scale Prediction

CRFs are good for cleaning up the CNN. We choose to use a CRF where every pixel pair is connected with an edge, rather than connecting each pixel only to its neighbors. We use this energy function (for pixel labeling $\mathbf{x}$)

$$E(\mathbf{x}) = \sum_i \theta_i(x_i) + \sum_i \sum_j \theta_{i,j}(x_i, x_j)$$

where $\theta_i(x_i) = -\log P(x_i)$ ($P(x_i)$ is label assignment probability of pixel i) and

$$\theta_{i,j}(x_i, x_j) = \mu(x_i, x_j) \sum_{m=1}^{K} w_m k^m(\mathbf{f}_i, \mathbf{f}_j)$$

where $\mu(x_i, x_j) = I(x_i \neq x_j)$, the $\mathbf{f}$ are extracted feature vectors, w_m is a weight, and k^m is a Gaussian kernel of the form:

$$w_1 \exp(-\frac{||p_i - p_j||^2}{2\sigma_\alpha^2} - \frac{||I_i - I - j||^2}{2\sigma_\alpha^2}) + w_2 \exp(-\frac{||p_i - p_j||^2}{2\sigma_\gamma^2})$$

where the p’s are pixel positions and the I’s are color intensities. You can optimize this energy function pretty efficiently.

In addition to the CRF, we leverage multiscale predictions. Basically, we consider the input image + the first four max-pooling layers and stick a 2-layer CNN (128 3×3 maps in the first layer, 128 1×1 maps in the second) on each and stack the results onto the feature maps outputted by the last layer of the CNN (so the softmax gets an addition $5 \times 128 = 640$ channels). We fix the other weights and only tweak the weights of this MLP.

7 Experimental Evaluation

PASCAL VOC has 20 classes and 1 background class. It has 1450 images in EACH (4350 total) of the train, validation and test set. With some additional annotations, we can increase the training set size to 10,500 images. Performance is measured as intersection-over-union averaged over the 21 classes.

We fine-tune the pretrained VGG-16 with minibatch SGD + momentum using cross entropy loss and scheduled learning rate drops. We then use 100 validation images and pick the hyperparameters of the CRF (e.g. the σ’s and w’s) with coarse-to-fine search.

The CRF gives a 4% boost, which is huge and the multiscale features give a 1.5% boost. They help flesh out the object boundaries, which is main advantage over competitors.

Our final system gets 71.6% mean intersection-over-union.