Learning Hierarchical Features for Scene Labeling

1 Citation

Farabet, Clement, et al. "Learning hierarchical features for scene labeling." IEEE transactions on pattern analysis and machine intelligence 35.8 (2013): 1915-1929.

https://hal-upec-upem.archives-ouvertes.fr/file/index/docid/742077/filename/farabet-pami-13.pdf

2 Abstract

We use a CNN feature extractor to build a system that can label each image in a pixel with its category. We try many postprocessing steps, one of which is a way to take a pool of segmentation components and select the best components to describe the image. We set state-of-the-art on the 33-class Sift Flow Dataset, the 170-class Barcelona dataset, and 8-class Stanford Background Dataset (near state-of-the-art). Our system processes a 320×240 image in under a second.

3 Introduction

Full-scene labeling (also called scene parsing) is when we tag every pixel with a category. There are two challenges in this problem: (1) making good feature vectors, (2) making sure that predictions are consistent with each other. We use a multi-scale convolutional neural network to label each pixel. The sophistication of the CNN allows our post-processing steps (which ensure consistency) to be relatively simple. Our two main components are:

1. CNN applied on different scales (from Laplacian pyramid) and trained to predict label for each pixel. It doesn't predict object boundaries, which is why we need post-processing.
2. We oversegment the image (i.e break it into regions), group feature vectors in each segment, and pass the result to labeling. Our main oversegmentation techniques are:
   (a) Superpixel: These are small regions that are roughly uniform within themselves. Each labeled pixel in the superpixel votes to label the superpixel. Superpixels are fixed-size, which is limiting.
   (b) Conditional Random Field over Superpixels: This graph-based approach smoothes the superpixels (avoids aberrations like person in the sky).
   (c) Multilevel Cut with Purity Criterion: Make family of segmentations at each level (e.g. segmentation tree, super-pixels with different algorithm parameters). For each pixel in a segment, get feature vector, and do component-wise max-pooling of feature vectors. Classify each segment and get probability distribution of classes. Select segments to minimize average impurity (i.e. entropy over class distribution)

All operations are nearly linear, with the CNN being the most compute-intensive. There are no parameters in the system.

4 Related Work

Previous work uses Markov Random Fields or Conditional Random Fields to smooth superpixels.

Other work uses deep learning on hand-engineered features.

Other work uses graph cuts to aggregate candidate segments (our purity criterion is easier).

5 Multiscale Feature Extraction for Scene Parsing

We want a feature extractor $f : \mathbb{R}^P \to \mathbb{R}^Q$ that turns an image $\mathbb{R}^P$ into a linearly-classifiable feature vector $\mathbb{R}^Q$. We do feature extraction with a multi-scale CNN. Given an image I, we create scaled versions X_s for $s \in \{1...N\}$ with a Laplacian pyramid (neighborhoods have zero-mean unit-variance) and process each densely with a CNN. Our CNN has three layers. The first two have convolution, *tanh* nonlinearity, and pooling, while the last only has convolution. That is:

$$f(X, \theta) = W_L H_{L-1}$$

$$H_l = \text{pool}(\tanh(W_l H_{l-1} + b_l))$$

where $L = 3$, $l \in \{1...L\}$, $H_0 = X$ and the W matrices apply convolution.

The output is a set of N feature maps $(\mathbf{f_1}, ..., \mathbf{f_N})$), which we upsample (denote as u) to have the same size. Then we concatenate them together to get the feature vector:

$$\mathbf{F} = [\mathbf{f_1}, u(\mathbf{f_2}), ..., u(\mathbf{f_N})]$$

To train the CNN, we strap on a softmax layer to predict the class distribution for pixel i as:

$$\hat{\mathbf{c}}_{i,a} = \frac{e^{\mathbf{w}_a^T \mathbf{F}_i}}{\sum_{b \in classes} e^{\mathbf{w}_b^T \mathbf{F}_i}}$$

and we optimize cross-entropy loss:

$$L_cat = \sum_{i \in pixels} \sum_{a \in classes} \mathbf{c}_{i,a} \ln \hat{\mathbf{c}}_{i,a}$$

6 Scene Labeling Strategies