Scene Parsing with Multiscale Feature Learning, Purity Trees, and Optimal Covers
Clément Farabet, Camille Couprie, Laurent Najman, Yann LeCun

Scene Parsing

The goal of Scene Parsing is to find a decomposition of the scene into objects (which are not necessarily disjoint).

It implies 3 sub-problems:
- Detection
- Recognition
- Delineation

Optimal (minimum entropy) cover of the pixels

Finding the purest labeling, i.e. for each pixel the parent component with the purest distribution is chosen.

Purity Tree

Using a hierarchy of segmentations and a class purity criterion, to decide if a segment contains one or more objects/classes.

Multi-scale Feature Learning

Using a multiscale convolutional network, trained to produce local & global features for region classification.

Overview of our Approach

why use a multiscale encoding?

- Each location is encoded by a multi-scale convolutional network: weights at each layer are replicated both spatially and in scale-space; this acts as a regularizer during training, as the filters learned need to be good for all scales.
- At each location, the resulting feature vector (768-dim) represents a local focus point as well as its context, up to the complete scene.

Feature learning procedure (pixelwise)

Convolutional network at each scale:

\[ H_k = F_{k-1} \times \mathcal{P} \]

Complete convolutional network (with \( \mathcal{P} \) an upsampling function):

\[ F = [F_1, u(F_2), \ldots, u(F_N)] \]

Classification loss:

\[ L_{\text{class}}(c_i, c_k) = \sum_{j \in \text{classes}} \mathcal{L}_j(c_i, c_k) \]

Optimal Cover

We produce an optimal cover of the pixels:

- Each component in the tree is now described by an estimate of the class histogram.
- Each pixel in the scene belongs to multiple pixel groupings in the tree.
- The one that has the lowest entropy is chosen to explain the pixel, and assign its class.
- A cover has interesting properties, as it is a set of non-disjoint components (allows for hierarchical classes).

Why use a multiscale encoding?

- Each location is encoded by a multi-scale convolutional network: weights at each layer are replicated both spatially and in scale-space; this acts as a regularizer during training, as the filters learned need to be good for all scales.
- At each location, the resulting feature vector (768-dim) represents a local focus point as well as its context, up to the complete scene.

Multi-scale Feature Learning

What's a purity tree?

- It is a tree.
- Its leaves are the pixels of the image.
- Its nodes are groups of connected pixels.
- Its root is the complete image.
- Each node is described by:
  - A cost (the purity of the node)
  - A histogram of classes present in the node

We compute the purity tree by:

- Computing a segmentation tree over the pixels (lots of standard methods to do that: plain gradient based, mean-shift, surface-normalized, …)
- Predicting the histogram of classes for each component
- Setting the “impurity” cost of each component as the entropy of that histogram

We predict class histograms:

- Each component is described by a grid of feature vectors:
  - A 3x3 grid of 768-dim feature vectors.
  - That yields a 7000-dim descriptor.
- Each descriptor is then projected through a 2-layer neural network, with 1024 hidden units, to produce an n-dim distribution (n classes):
  - This network is trained using KL-divergence, to approximate the histogram of classes present in each component.

Purity Tree

Results

<table>
<thead>
<tr>
<th>Method</th>
<th>8-class problem</th>
<th>20-class problem</th>
</tr>
</thead>
<tbody>
<tr>
<td>Goal</td>
<td>76.4</td>
<td>76.9</td>
</tr>
<tr>
<td>Mean</td>
<td>76.9</td>
<td>76.1</td>
</tr>
<tr>
<td>Median</td>
<td>79.4</td>
<td>79.4</td>
</tr>
<tr>
<td>Lempits et al.</td>
<td>81.9</td>
<td></td>
</tr>
<tr>
<td>Multiscale Cover + Tree</td>
<td>79.5</td>
<td>79.5</td>
</tr>
</tbody>
</table>

Table showing accuracy and runtime for different methods.

[Results Table]

[Diagram of a purity tree]