the recall does not drop below $r - \rho$ for some small ρ. This dictates an efficient algorithm for optimizing α: start with bracketing α between 0 and some large number, and perform binary search. We use two subsets of images: $\mathcal{I}_{train}$ is used to learn $\mathbf{w}^*(\alpha)$ in each iteration of the binary search as per 1, while $\mathcal{I}_{tune}$ is used to evaluate average recall obtained with $\mathbf{w}^*(\alpha)$ on images in $\mathcal{I}_{tune}$.

4. Cascaded Region Agglomeration

When the model is trained with the scaled loss (1), with α optimized to limit recall drop on tuning set, the merging usually stops early, with many remaining unmerged regions. This is in part due to the very "cautious" model learned with the scaled loss, and in part to the fact that once many of the regions in the initial segmentation are merged, and the features of their surviving neighbors are updated, the distribution of the features no longer matches the one on which we trained Pg. Color and texture histograms tend to become less sparse; shape may become less convex; features that were useless for very small regions (e.g., counts of visual words) may become useful for larger regions, etc. This observation leads to a simple idea: re-train the model on the new, larger regions. The second model merges some more segments, but then it also stops. We can then train a third model, etc., as described below.

Algorithm 2: Cascaded Segmentation

```
Given: Image $I$, initial $\mathcal{R}$,weights $\mathbf{w}_1, \ldots, \mathbf{w}_T$
$\mathcal{R}_0 \leftarrow \mathcal{R}$
for $t = 1$ to $T$ do $R_t \leftarrow \textsc{Merge}(I, \mathcal{R}_{t-1}, \mathbf{w}_t)$
Return: $\mathcal{R}_T$
```

Training the cascade (Algorithm 3) is similar to the training of cascaded classifiers elsewhere, e.g., in the Viola-Jones face detector [23]. It is also similar in spirit to the re-training of pixel merge likelihood in [11]. One important difference from these is that we use asymmetric loss, rather than tune the threshold on classification. This enables us to train a deeper cascade, and helps performance as we show in Section 5.

Algorithm 3: Training a cascade

```
Given: $\{I_i, \mathcal{R}_i, G_i\}_{i=1}^N,\ \rho > 0,\ T$
for $t = 1$ to $T$ do
    sample image subsets $\mathcal{I}_{train}, \mathcal{I}_{tune}$,
      s.t. $\mathcal{I}_{tune} \cap \mathcal{I}_{train} = \varnothing$
    Find $\alpha_t^*$ with binary search, using $\mathcal{I}_{train}, \mathcal{I}_{tune}, \rho$
    $\mathbf{w}_t \leftarrow \mathbf{w}^*(\alpha_t^*)$ by Eq. (1)
    foreach $i \in \{1, \ldots, N\}$ do
        merge $R_i \leftarrow \textsc{Merge}(I_i, \mathcal{R}_i, \mathbf{w}_t)$
Return: $\mathbf{w}_1, \ldots, \mathbf{w}_T$
```

At each stage of Algorithm 3, $\mathcal{I}_{train}$ and $\mathcal{I}_{tune}$ are mutually exclusive, to prevent overfitting of α. Furthermore, at each stage these two sets are sampled independently, so that empirical distribution of features on with stage t is a more robust estimate of the distribution of new data. Finally, note that after a few stages most of the boundaries from earlier stages are no longer active (due to merging of their constituent regions) and so reusing the same images carries much less risk of overfitting.

To summarize: ISCRA starts with an initial set of small superpixels, and propagates them through a series of stages. At each stage some of the regions are merged using the model learned for that stage, and the next stage receives the resulting coarsened segmentation as its input.

Once the merging stops, we can "backtrack" and report the segmentation at any point along the merging process. This allows us to control the scale either by specifying the desired number of segments (appropriate for the superpixel regime) or by specifying the number of stages to run, which is the natural definition of scale for ISCRA. We can also compute the boundary map that reflect the scale at which regions are merged: if there are T stages in ISCRA, then for every boundary pixel in the initial segmentation, the value of the boundary map will be t/T if it was merged after t stages. Pixels that were not on the boundaries of initial superpixels will have values zero, and pixels that survived the last stage will have value 1. Examples of ISCRA segmentations at multiple scales, as well as the hierarchical boundary map, are illustrated in Figure 2(right); more examples are available in supplementary materials.

5. Experiments

Our experiments were aimed at two goals: (i) compare ISCRA to other methods in both superpixel and large region regimes; (ii) evaluate effect of various design choices on performance. Below we describe a few remaining implementation details, the experimental setup and the results.

5.1. Features and training

The features we use can be grouped into three sets:

Appearance features that measure difference in the "content" of the two regions. These include

- **Color**: The χ^2 difference of color histograms, computed for each channel in L*a*b color space, with 32 bins per channel. This yields 3 dimensions in ϕ.
- **Texture**: The χ^2 difference of two segments when representing each image using 32 textons (1 dimension).
- **Geometric Context**: For each of seven geometric context labels [10], compute χ^2 difference between histograms of values for that label within the regions, with 32 bins (7 dimensions).