AUTOMATIC IMAGE SEGMENTATION VIA DYNAMIC REGION MERGING

Team Noisy Pixel

Shubham Dokania (2020701016)
Shanthika Shankar Naik (2020701013)
Sai Amrit Patnaik (2020701026)

SEGMENTATION WITH REGION GRAPHS

Segmentation is a difficult problem to solve in image processing and computer vision, yet contains high potential for image understanding, analysis, and is useful in several downstream tasks.

- For this paper, we start with an initially over-segmented image with many regions (can contain vertices or super-pixels).
- Formulate the image as a graph and segmentation over this graph as an optimization problem.
- Iteratively merge regions according to some statistical tests.
- Build the objective function and propose a solution based on dynamic programming.
- Show that local segmentation using this method also satisfies global segmentation properties.

Building the Region Graph

- Given an image, we consider the it in the form of a graph G = (V, E), with V as the set of vertices and E as the set of edges.
- The graph G is an undirected graph, and the collection of vertices represent pixels or regions. The edges connect these regions which are adjacent to each other.
- Each edge has an associated weight which is given by minimum dissimilarity between two regions.

CONDITIONS (REGION MERGING)

- Region Merging is essentially a two part process:
  - First, we need to check whether two regions are compatible with each other to be merged or not.
  - Second, we need to decide a stopping criterion, otherwise all regions will eventually end up being merged together.
- For region merging, the weights over edges of the graph are considered. The regions with are most similar among themselves are merged together.
- For a stopping test, we use the Sequential Probability Ratio Test (SPRT) proposed by Wald.
- These tests take into account, the visual cues present in the image and the region to make a decision for merging.

$$P(R_1, R_2) = \begin{cases} true & \text{if (a) } S(R_1, R_2) = \min_{R_i \in \Omega_1} S(R_1, R_i) = \min_{R_j \in \Omega_2} S(R_2, R_j)\text{: and} \\ & \text{(b) } R_1 \text{ and } R_2 \text{ are consistent} \\ false & \text{otherwise} \end{cases}$$

TEST CONSISTENCY

We assume a parameter θ be related to the distribution of random cues in the image i.e. we gather information about the parameter by observing the random variable in multiple successive steps by hypothesis testing.

We form a null hypothesis for the evaluation task that whether two regions are “consistent” or “inconsistent”, which are denoted by the null hypothesis and the alternate hypothesis.

The property of the hypothesis is a hidden state which is not directly observable but is statistically linked to the observable cues.

We make use of the *SPRT* (Sequential Probability Ratio Test) to make successive assumptions about the regions to arrive at value for the parameters which then helps in deciding which hypothesis to select.

We observe the distribution of random cues in a sequential manner until a likelihood ratio for hypothesis testing goes out of a predefined interval (B, A).

$$\delta_i = \log \frac{P_1(x_i \mid \theta_1)}{P_0(x_i \mid \theta_0)}, \quad i = 1,2,\ldots,N$$

$$\begin{cases} P_0(x \mid \theta_0) = 1 - \lambda_1 \exp(-(I_b - I_a) S_I^{-1} (I_b - I_a)) \\ P_1(x \mid \theta_1) = 1 - \lambda_2 \exp(-(I_b - I_{a+b}) S_I^{-1} (I_b - I_{a+b})) \end{cases}$$

The final likelihood for testing is the sum of individual ratio tests. We terminate the test in case any of the range constraint is violated

RAG & MERGING

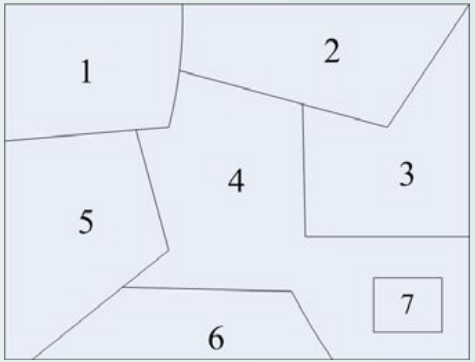
Algorithm for consistency test

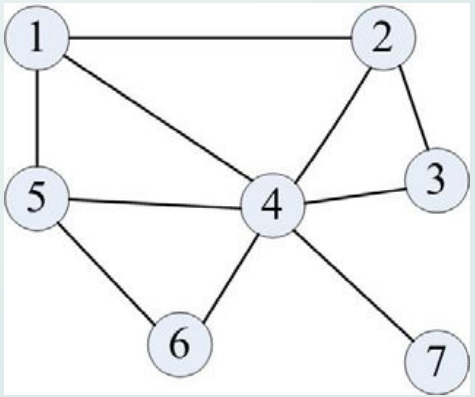
Preset λ_1;
Let λ_2=1, α=0.05, β=0.05;
Compute parameters:
N_0: be a constant greater than $\max\{E\{\delta|\theta_0\}, E\{\delta|\theta_1\}\}$;
$A=\log(1-\beta)/\alpha$, $B=\log\beta(1-\alpha)$;
$P_0(x|\theta_0)$, $P_1(x|\theta_1)$ are computed using Eq. (4).

Input: a pair of neighboring regions.
Output: the decision D that the two regions are "consistent" (D=1) or "inconsistent" (D=0).

1. Set evidence accumulator δ and the trials counter n to be 0.
2. Randomly choose m pixels in each of the pair of regions, where m equals the half size of the region.
3. Calculate the distributions of visual cues x using Eq. (4) based on these pixels.
4. Update the evidence accumulator $\delta = \delta + \log\frac{P_1(x|\theta_1)(1-\beta)}{P_0(x|\theta_0)(1-\alpha)}$.
5. If $n \le N_0$
   If $\delta \ge A$, return D=1 (consistent)
   If $\delta \le B$, return D=0 (inconsistent)
   If $n > N_0$
   If $\delta \ge 0$, return D=1 (consistent)
   If $\delta < 0$, return D=0 (inconsistent)
6. Go back to step 2.

Region Adjacency Graph formation from parts of Image





DYNAMIC REGION MERGING

- As stated before, we start with an over-segmented image. The reason is that small region provide more details about the structure of local areas in an image.
- We can use watershed, SLIC, or K-means based simple methods to obtain such an initial segmentation easily.
- Considering the region merging as a labelling problem, we see that each region in the graph changes from an initial label to a new label on each iteration (could be same or repetitive as well), and we want to find the overall final label for these regions.
  In this setup, the final label is not unique due to the random sampling in SPRT test.
- In the process of merging, the label of each region is sequentially transited from initial to the final one; forming a sequence.

OPTIMIZATION PROBLEM

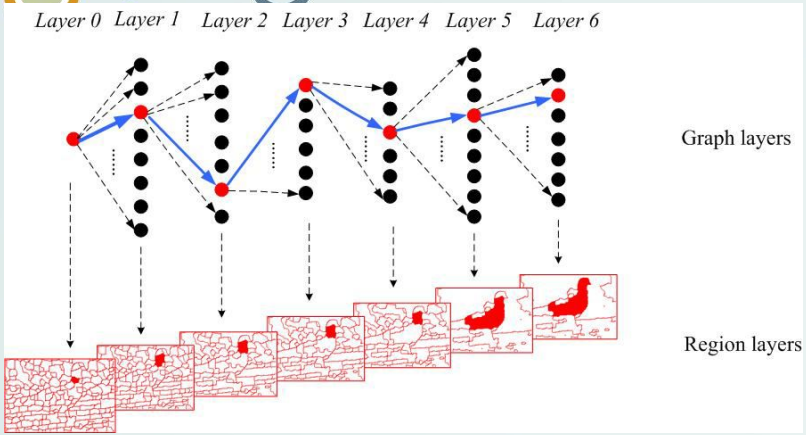
Ideally, we want to find the optimal sequence of merges which produces the union of optimal labeling for all the regions, and hence minimizes the cost of such transitions.

$$\begin{aligned}\min F_i(l_i^0,\cdots,l_i^n) &= \min F_i(l_i^0,l_i^{n-1}) + d_{n-1,n} \\ &= \min F_i(l_i^0,l_i^{n-2}) + d_{n-2,n-1} + d_{n-1,n} \\ &= \cdots \\ &= \sum_{k=0}^{n-1} d_{k,k+1}\end{aligned}$$

We define the cost for one region, based on the above assumptions as shown (left). Here, each parameter in the final sum is the cost of transition over two labels in the same region.

To make things simpler, we can define this as a dynamic problem, as stated in the equation above.

Dynamic Programming Problem



We can assume each transition as a layer in a graph, and the weight on the path to be the cost. This way, the changes happening across the entire RAG through time becomes a big graph problem where we need to find the shortest path.

And for this task, the dijkstra's algorithm is taken into application.

Overall algorithm as a block

Input: the initially over segmented image S_0.

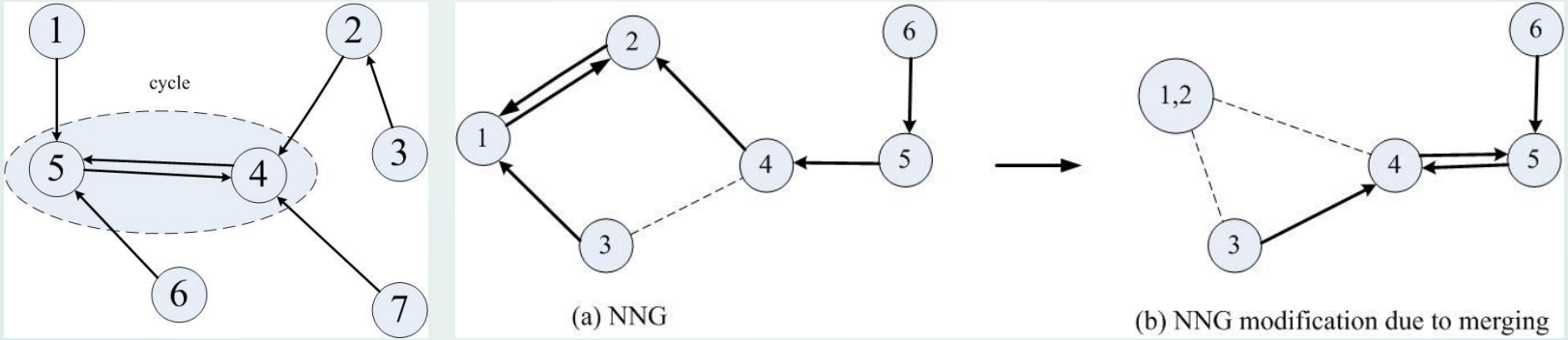
Output: region merging result.

1. Set i=0.
2. For each region in segmentation S_i, use **Algorithm 1** to check the value of predicate P with respect to its neighboring regions.
3. Merge the pairs of neighboring regions whose predicate P is true, such that segmentation S_{i+1} is constructed.
4. Go back to step 2 until $S_{i+1} = S_i$.
5. Return S_i.

NEAREST NEIGHBOUR GRAPH

It is easy to see that the previous algorithm has a high time complexity due to the evaluation of the minimum weights between regions for every change and requires a search over all edges at each iteration.

Such an evaluation is very slow and even for regions which have been merged, it becomes a redundant operation. To cope with this problem, the authors propose a variation to the graph which is formed by creating a Nearest neighbour alternative called the NNG (Nearest neighbour graph) *(left)*.



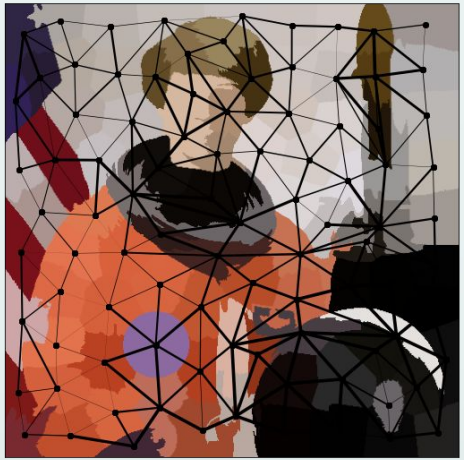
The nearest neighbour graph approach *(right)* follows the below algorithm by merging the region nodes into one and keeping track of the weight changes. Due to this, we also notice some desirable features in the search for new regions to be merged as shown below:

CURRENT PROGRESS

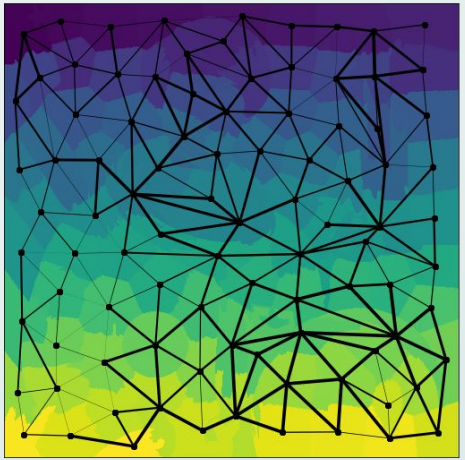
1. Finished multiple iterations of **paper reading** and discussions over the core idea in the report and prepared plans (with milestones) for a working prototype.
2. Implemented **watershed algorithm** from scratch that gives the initial over segmented regions of the input image. The implementation can be found here.
3. Completed the code for creating the **RAG** from the segmentation labels. The implementation can be found here.
4. Initial model for **SPRT** and hypothesis testing prepared, currently reviewing the code for minor patches.

CURRENT RESULTS:

Original Image



Segmented Image with RAG



Label Map with RAG

THANKS

Do you have any questions?

CREDITS: This presentation template was created by **Slidesgo**, including icons by **Flaticon**, and infographics & images by **Freepik**