

Ruolin Li
31764160

CPSC 320: Clustering *

You're working on software to organize people's photos. Your algorithm receives as input:

- A bunch of uncategorized photos.
- A *similarity measure* for each pair of photos, where a 0 similarity indicates two photos are nothing like each other; a 1 indicates two photos are exactly the same. All other similarities are in between.
- The number of categories to group them into.

Your algorithm should create a *categorization*: the requested number of categories, where a category is a non-empty set of photos. Every photo belongs to some category, and no photo belongs to more than one category. So, a categorization is a *partition*. We'd like similar photos to be in the same category.

Step 1: Build intuition through examples.

1. Write down small and trivial instances of the problem. What data structure is useful to represent a problem instance? Write down also potential solutions for your instances. Are some solutions better than others? How so?

trivial instance: An empty graph.
Let n be the # of photos and c be the number of categories requested. Then when $c=1$ it is also a trivial case because every photo is going to one category.
② When $c=n$, then one photo belongs to one category.

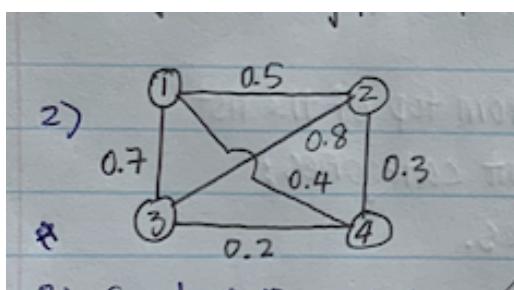
*Copyright Notice: UBC retains the rights to this document. You may not distribute this document without permission.

Step 2: Develop a formal problem specification

1. Develop notation for describing a problem instance.

The graph is complete. n : # of photos/a graph with n nodes, $\{1, 2, \dots, n\}$.
1) $G = (V, E)$ $C = N_0$ we have (v_1, v_2, d) to describe an edge between v_1 and v_2 and d represents the similarity. $v_1, v_2 \in V$, $d \in [0, 1]$.
One edge for every pair of nodes, and assign the similarities to those edges.

2. Use your notation to flesh out the following group of photos into an instance.



3. Develop notation for describing a potential solution. Describe what you think makes a solution **good**. Can you come up with a reasonable criterion for deciding if one solution is better than another?

3) Good solution: Let range (C_i) be max similarity ~ min similarity between two photos in C_i . Minimize average range, or maximize avg range.

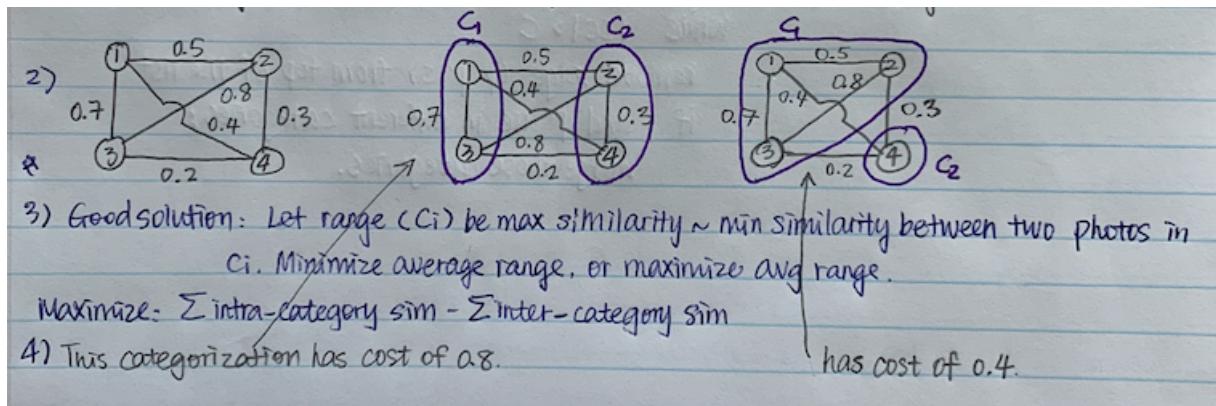
$$\text{Maximize: } \sum \text{intra-category sim} - \sum \text{inter-category sim}$$

4. From here on, we'll all use the same definition of "good solution".

First, we define the similarity between two categories C_1 and C_2 to be the maximum similarity between any pair of photos p_1, p_2 such that $p_1 \in C_1$ and $p_2 \in C_2$.

Then, the **cost** of a categorization is the maximum similarity between any two of its categories. The lower the cost, the better the categorization, since we don't want categories to be similar. So, we want to find a solution with minimum cost. We'll use the term "optimal solution" (rather than "good solution") to refer to solutions that have minimum cost.

Write down optimal solutions and their costs for your previous examples.



Step 3: Identify similar problems. What are the similarities?

It is similar to weighted graph problem, minimum spanning tree problem.

Step 4: Evaluate brute force.

1. A potential solution is the set of partitions of n photos into c subsets (where c is the requested number of categories). Suppose that $c = 2$. Roughly, how does the number of potential solutions grow asymptotically with n ? Polynomially? Exponentially?

Brute Force

1. n photos, 2 categories C_1, C_2 . # of different possibilities.
 C_1 is a subset of $\{1, 2, \dots, n\}$, this set has 2^n subsets but can't be \emptyset or $\{1, 2, \dots, n\}$
 C_1 has $(2^n - 2)$ possibilities.
and $\{C_1, C_2\}$ is equal to $\{C_2, C_1\}$, so same applies to C_2 .
Total solutions: $\frac{1}{2}(2^n - 2) \quad \Theta(2^n)$

2. Given a potential solution, how can you determine how good it is, i.e., what is its cost? Asymptotically, how long will this take?

2. We need to figure out max inter-group similarity; max cost of an inter-category edge.
set max to 0. for each edge $(p, p', s) \leftarrow \Theta(n^2)$
if (p, p') is intercategory edge and $s > \text{max} \leftarrow \Theta(1)$
 $\text{max} = s$.

Total runtime $\Theta(2^n \cdot n^2)$. ① + ②

Step 5: Design a better algorithm.

There is a much better approach.

1. Find the edge in each of your instances with the highest similarity. Should the two photos incident on that edge go in the same category? Prove a more general result.

Yes. For any problem instance, there's an optimal solution in which 2 photos, p & p' , with highest similarity, are placed in the same category.
If T is a solution in which p, p' are in different categories, then the cost of T is the similarity of p & p' .
Any solution S , including any solution putting p & p' together, has a cost \leq similarity of p & p' .

2. Based on this insight, come up with algorithmic ideas for creating a categorization.

2. Procedure Clustering - Greedy

create a list of edges in decreasing order of similarity.

initial categories: each photo belongs to its own category.

while $|C| > c$

remove edge (p, p', s) from top of the list

if p and p' are in different categories;

merge those categories.