CS131
Foreground-Background Segmentation via Clustering

Yuke Zhu
November 7, 2014
Overview

• Use clustering algorithms to segment images
• Evaluate by cats out of images
• Not a lot of code! (< 100 lines)
• Focus on experimentation
Overall Flow

Prerequisite: Implement clustering algorithms
Input: an image

1. Compute a feature vector for each pixel
2. Cluster the feature vectors
3. Assign pixels to segments based on the clusters
4. Choose some subset of segments as “foreground”
5. Transfer foreground to another image
6. Compare foreground with ground truth
Overall Flow

Prerequisite: Implement clustering algorithms
Input: an image
1. Compute a feature vector for each pixel
2. Cluster the feature vectors
3. Assign pixels to segments based on the clusters
4. Choose some subset of segments as “foreground”
5. Transfer foreground to another image
6. Compare foreground with ground truth
You need to implement 2 clustering methods:

- **K-Means Clustering**
  - KMeansClustering.m
  - Covered in Lecture 13

- **Hierarchical Agglomerative clustering**
  - HAClustering.m
  - Covered in Lecture 12

Section 2 of assignment
Clustering Algorithms: Interface

function idx = KMeansClustering(X, k, visualize2D)
function idx = HAClustering(X, k, visualize2D)

X: Matrix where each row is a point
k: Number of clusters
visualize2D: If clustering 2D points, set this to true to see a visualization
idx: Vector giving computed assignments of points to clusters

Use KMeansClusteringTest and HAClusteringTest to validate your implementation.

Note: KMeansClustering has an additional parameter centers that you don’t need to worry about
K-Means Clustering
K-Means Clustering
K-Means Clustering
K-Means Clustering
Hierarchical Agglomerative Clustering
Hierarchical Agglomerative Clustering
Hierarchical Agglomerative Clustering
Hierarchical Agglomerative Clustering
Clustering: Efficiency Matters!

- For loops are SLOW in MATLAB; avoid them wherever possible!
- Useful MATLAB functions: min, mean, pdist2, ind2sub, randperm
- HAClustering.m:
  - Your code can be written with no for loops
  - For reference: clustering 5000 5D points on my laptop takes about 90 seconds
- KMeansClustering.m
  - For reference: clustering 5000 5D points on my laptop takes < 1 second
Overall Flow

Prerequisite: Implement clustering algorithms
Input: an image

1. Compute a feature vector for each pixel
2. Cluster the feature vectors
3. Assign pixels to segments based on the clusters
4. Choose some subset of segments as “foreground”
5. Transfer foreground to another image
6. Compare foreground with ground truth
Pixel Feature Vectors

You need to at least two types of features:

• Color features: (r, g, b)
  - Done for you
  - `ComputeColorFeatures.m`

• Color and position features: (r, g, b, x, y)
  - You need to implement this
  - `ComputePositionColorFeatures.m`

Test with `ComputePositionColorFeaturesTest.m`
Pixel Feature Vectors: Normalization

• Normalization is applied to feature vectors before clustering as a preprocessing step
• There are many types of normalization
• For this assignment we will normalize each feature to have **zero mean** and **unit variance**: 

\[
\mu_j = \frac{1}{n} \sum_{i=1}^{n} f_{ij} \quad \quad \quad \quad \sigma_j^2 = \frac{1}{n-1} \sum_{i=1}^{n} (f_{ij} - \mu_j)^2 
\]

\[
\tilde{f}_{ij} = \frac{f_{ij} - \mu_j}{\sigma_j} 
\]

NormalizeFeatures and NormalizeFeaturesTest
Pixel Feature Vectors: Extra Credit

Implement your own feature vectors and see how they perform

Some ideas:
• Gradients
• Edges
• SIFT descriptors

Use ComputeFeatures.m as a starting point
function features = ComputeColorFeatures(img)
function features = ComputePositionColorFeatures(img)

img: h x w x 3 matrix of pixel data for image
features: h x w x d matrix of features for each pixel

Any custom feature vectors you write should have the same interface!
Overall Flow

Prerequisite: Implement clustering algorithms

Input: an image

1. Compute a feature vector for each pixel
2. Cluster the feature vectors
3. Assign pixels to segments based on the clusters
4. Choose some subset of segments as “foreground”
5. Transfer foreground to another image
6. Compare foreground with ground truth
Cluster Feature Vectors + Assign Pixels

• This is done for you in `ComputeSegmentation.m` and `MakeSegments.m`

• `ComputeSegmentation.m` has many tunable parameters - read the documentation in the file!

• The data structure used to store a segmentation is described in `MakeSegments.m`

• Use `RunComputeSegmentation.m` as a starting point for your custom feature vectors
Resizing to Speed up Segmentation

Implemented for you in ComputeSegmentation.m
- just set the resize parameter!
Overall Flow

Prerequisite: Implement clustering algorithms
Input: an image
1. Compute a feature vector for each pixel
2. Cluster the feature vectors
3. Assign pixels to segments based on the clusters
4. Choose some subset of segments as “foreground”
5. Transfer foreground to another image
6. Compare foreground with ground truth
Choose Foreground Segments

• After segmenting an image, foreground object may be split across several segments
• Use ChooseSegments.m to pick a subset of segments as foreground
Overall Flow

Prerequisite: Implement clustering algorithms
Input: an image
1. Compute a feature vector for each pixel
2. Cluster the feature vectors
3. Assign pixels to segments based on the clusters
4. Choose some subset of segments as “foreground”
5. Transfer foreground to another image
6. Compare foreground with ground truth
Transfer Foreground

• Just use `ChooseSegments.m` but pass in a background image
Overall Flow

Prerequisite: Implement clustering algorithms
Input: an image
1. Compute a feature vector for each pixel
2. Cluster the feature vectors
3. Assign pixels to segments based on the clusters
4. Choose some subset of segments as “foreground”
5. Transfer foreground to another image
6. Compare foreground with ground truth
Compare with Ground Truth

• We provide a small dataset of 17 pictures of cats with correct segmentations
• The accuracy of a segmentation is the fraction of pixels that are correctly labeled as foreground / background
• EvaluateSegmentation.m computes the accuracy of a segmentation
• Use EvaluateAllSegmentations.m as a starting point to evaluate your method on all images in the dataset (> 0.8)
Compare with Ground Truth

Original image

Ground truth segmentation

Kmeans segmentation
Accuracy = 0.8877

HAC segmentation
Accuracy = 0.8784
What to do in writeup

• Answer all questions from the “In your Writeup” sections

• Focus on experimentation
  - Vary the segmentation parameters: feature transform, feature normalization, clustering algorithm, number of clusters, resize
  - How do your results change? (Qualitatively and quantitatively)

Due: 2014.11.21 5:00pm