Fast, Robust and Scalable Clustering Algorithms

with Applications in Computer Vision

Vahan Petrosyan

Department of Automatic Control, KTH Royal Institute of Technology

Supervisor: Alexandre Proutiere, Co-supervisor: Mikael Johansson

30 November 2018

Outline

- Introduction to Clustering Algorithms
- Motivation
- Cluster Validation
- Cluster Initialization
- Density Transformation
- Clustering Pixels in the Images
  - Application: Image Annotation

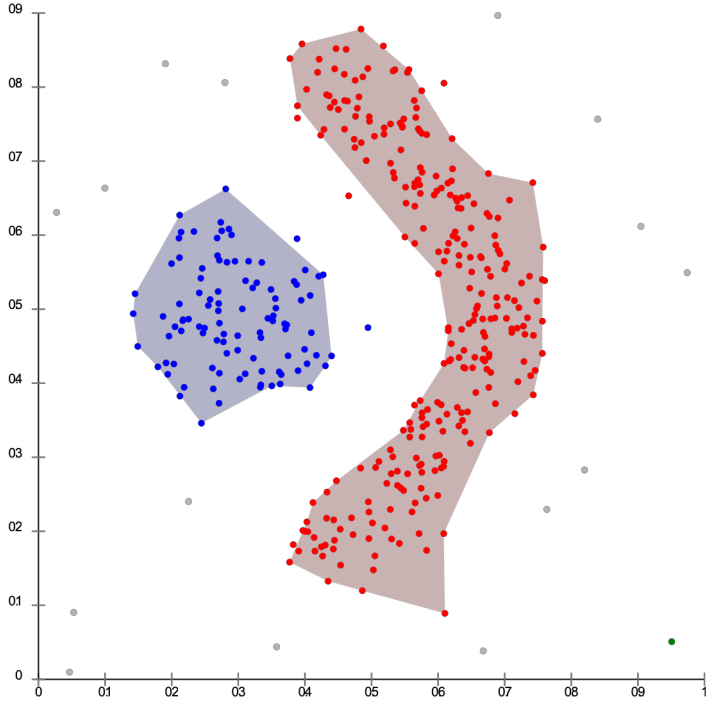
Outline

- Introduction to Clustering Algorithms
- Motivation
- Cluster Validation
- Cluster Initialization
- Density Transformation
- Clustering Pixels in the Images
  - Application: Image Annotation

Clustering

- **Clustering** is the task of grouping a set of objects such that more “similar” object will appear in the same group

- Applications:
  - Document Classification
  - Insurance Fraud Detection
  - Gene Expression
  - Image Segmentation
  - etc.



Clustering Algorithms

- K-means
- Spectral Clustering
- Level Set Clustering
- Hierarchical Clustering
- Density-Based Clustering
- Mode-Based Clustering
- Expectation Maximization
- etc.

Clustering Algorithms

- K-means
- Spectral Clustering
- Level Set Clustering
- Hierarchical Clustering
- Density-Based Clustering
- Mode-Based Clustering
- Expectation Maximization
- etc.

K-means

- Objective: $$J(\mathcal{C}) = \sum_{j=1}^{k} \sum_{x_i \in c_j} ||x_i - \mu_j||^2$$

- Algorithm

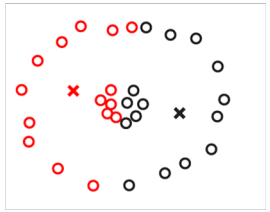
K-means

1. Select an initial partition with k clusters.
2. Repeat steps 3 and 4 until cluster membership stabilizes.
3. Generate a new partition by assigning each pattern to its closest cluster center.
4. Compute new cluster centers.

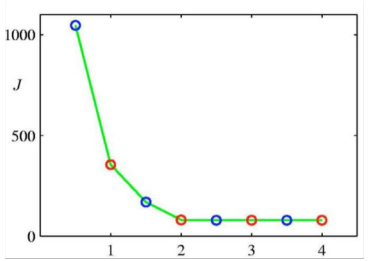
K-means cont.

- Parameters
  - Number of Clusters, k
  - Cluster Initialization
  - Distance Metric
- Drawbacks

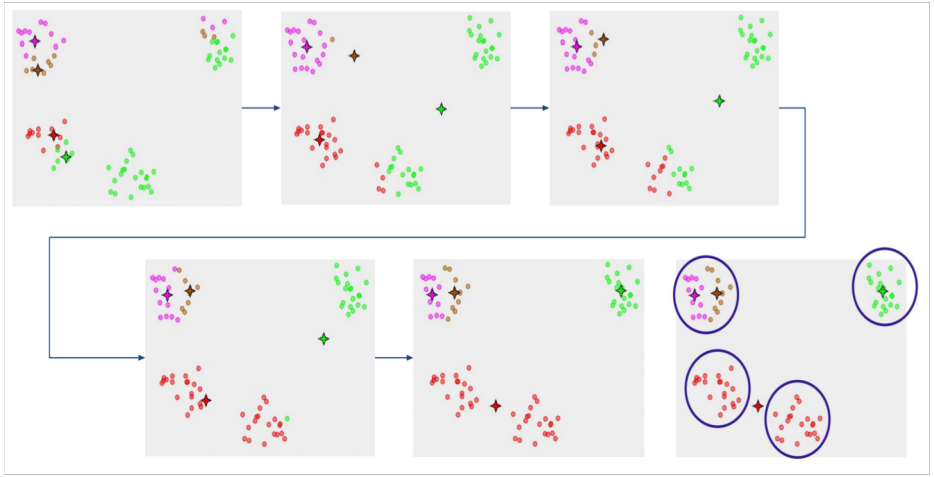
1. Finds only center based clusters



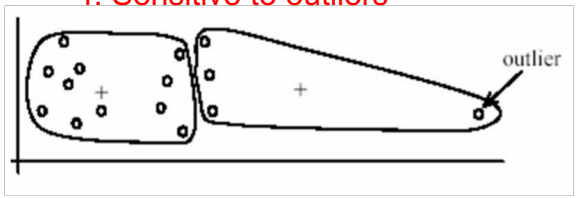
2. k is not known



3. Sensitive to initialization



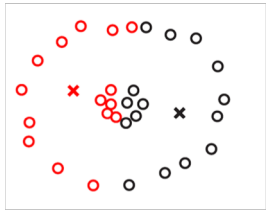
4. Sensitive to outliers



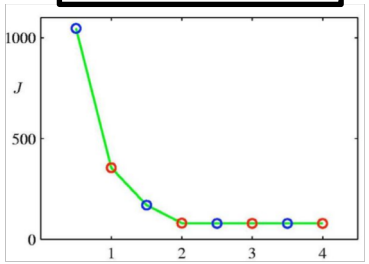
K-means cont.

- Parameters
  - Number of Clusters, k
  - Cluster Initialization
  - Distance Metric
- Drawbacks

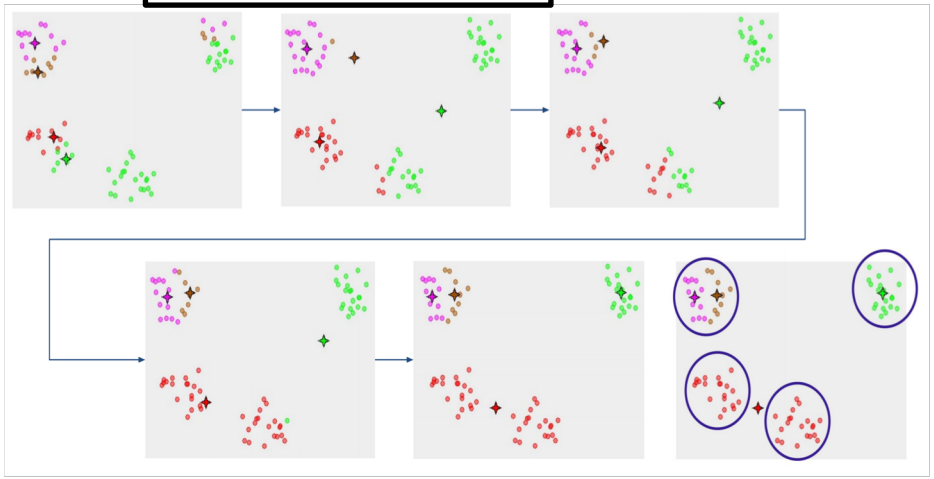
1. Finds only center based clusters ✗



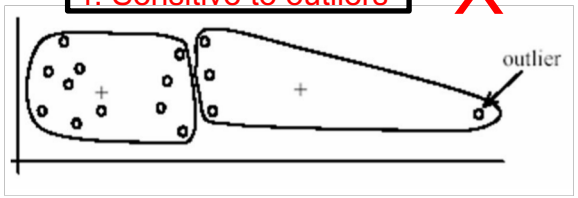
2. k is not known ✓



3. Sensitive to initialization ✓



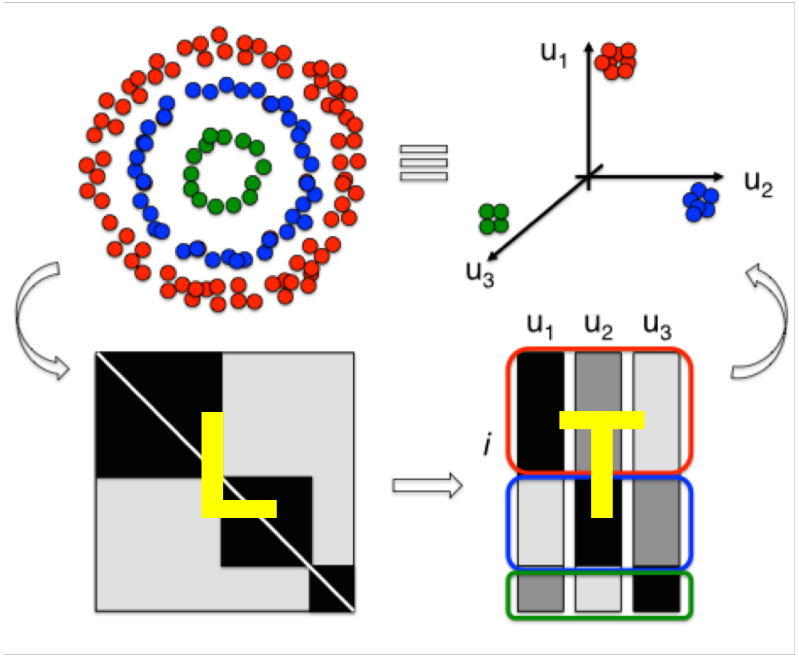
4. Sensitive to outliers ✗



Spectral Clustering

Spectral Clustering

1. Input dataset X and select the number of clusters k.
2. Construct S
3. Compute normalized Laplacian L
4. Compute the first k eigenvectors $u_1, \cdots, u_k$ of L
5. Let $U \in \mathbb{R}^{n \times k}$ be the matrix containing the vectors $u_1, \cdots, u_k$ as columns
6. Form the matrix $T \in \mathbb{R}^{n \times k}$ from U by normalizing the rows to norm 1
7. Use the matrix T as an input to k-means algorithm
8. Set output of k-means as the cluster assignment of input data X

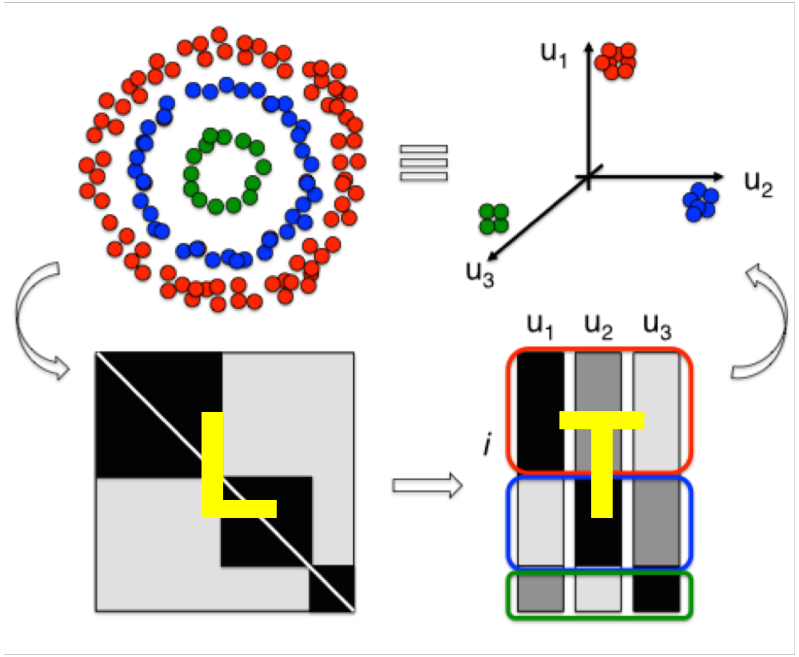


where $S_{i,j} = e^{\frac{|x_i - x_j|^2}{\sigma^2}}$

Spectral Clustering

Spectral Clustering

1. Input dataset X and select the number of clusters k.
2. Construct S
3. Compute normalized Laplacian L
4. Compute the first k eigenvectors $u_1, \cdots, u_k$ of L
5. Let $U \in \mathbb{R}^{n \times k}$ be the matrix containing the vectors $u_1, \cdots, u_k$ as columns
6. Form the matrix $T \in \mathbb{R}^{n \times k}$ from U by normalizing the rows to norm 1
7. Use the matrix T as an input to k-means algorithm
8. Set output of k-means as the cluster assignment of input data X



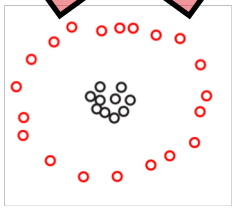
where $S_{i,j} = e^{\frac{|x_i - x_j|^2}{\sigma^2}}$

Spectral Clustering cont.

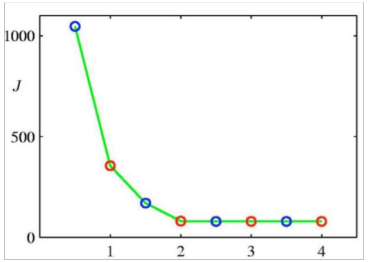
- Parameters
  - Same as K-means
  - σ

- Drawbacks

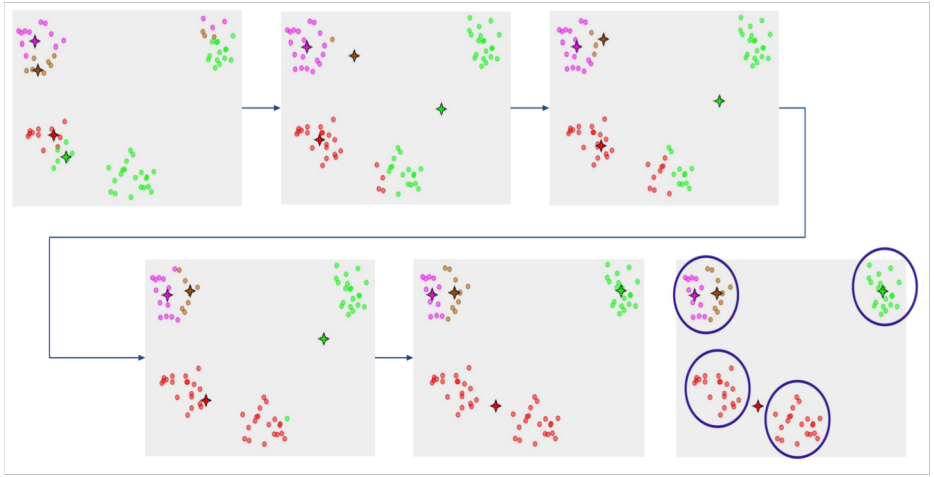
1. Finds only center based clusters



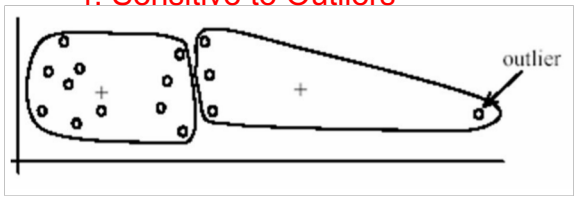
2. k is not known



3. Sensitive to Initialization



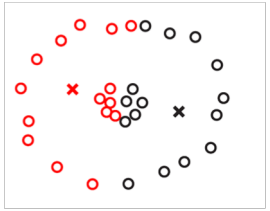
4. Sensitive to Outliers



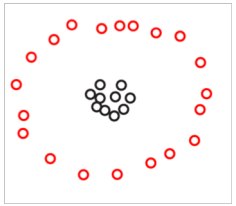
Spectral Clustering cont.

- Parameters
  - Same as K-means
  - σ
- Drawbacks

Untuned σ



Tuned σ

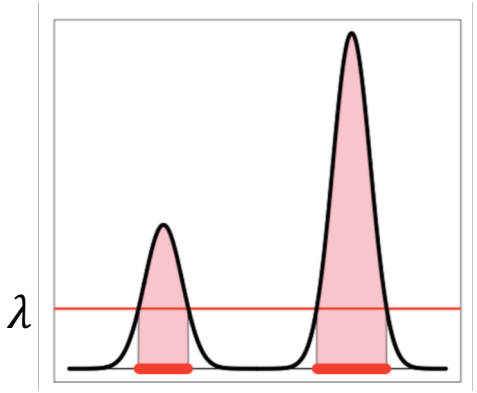


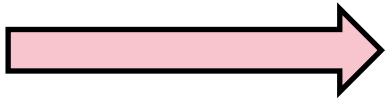
Tuning σ is a challenging problem

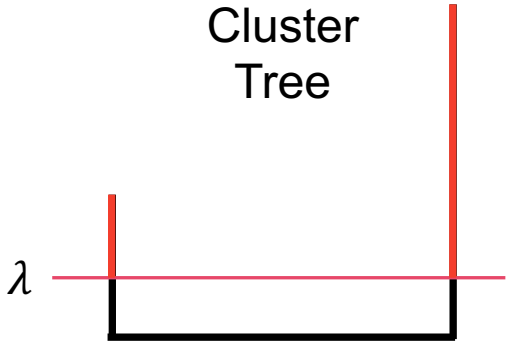
Level Set Clustering

- Given: $X = \{x_1, \cdots, x_n\}$ i.i.d. sample drawn from $p(x|\theta)$
- Upper Level Set $L_\lambda(p) = \{x \in \mathbb{R}^d : p(x|\theta) \geq \lambda\}.$
- Connected components of $L_\lambda(p)$ form a cluster at each level λ
- Challenge: The estimation of $p(x|\theta)$ is not trivial

Level Set Clustering cont.







$$L_\lambda(p) = \{x \in \mathbb{R}^d : p(x|\theta) \geq \lambda\}.$$

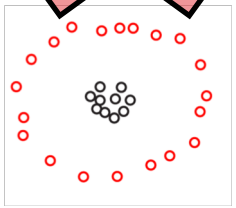
Visualize Clusters with Cluster Trees

Level Set Clustering cont.

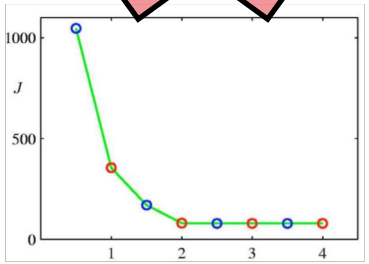
- Parameters
  - Parameters for estimating $\hat{p}(x|\theta)$
  - λ

- Drawbacks

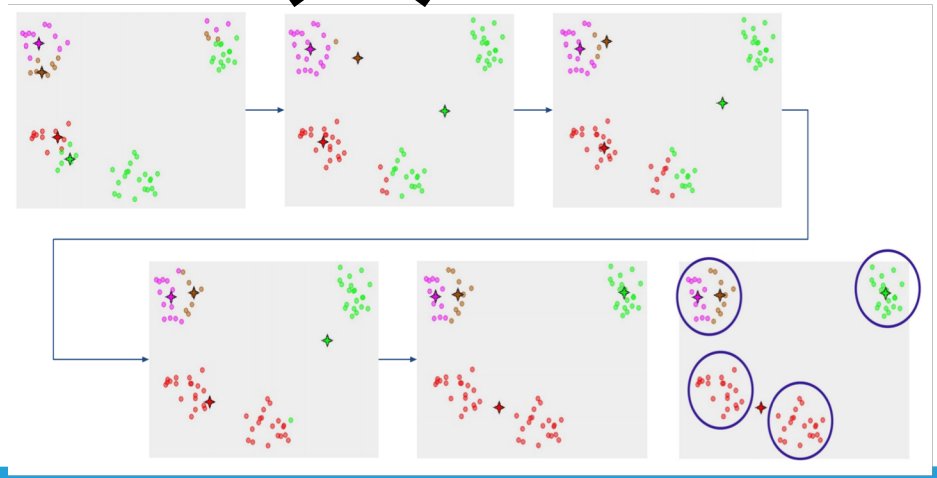
1. Finds only center based clusters



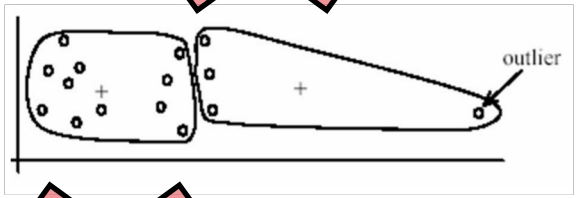
2. k is not known



3. Sensitive to Initialization



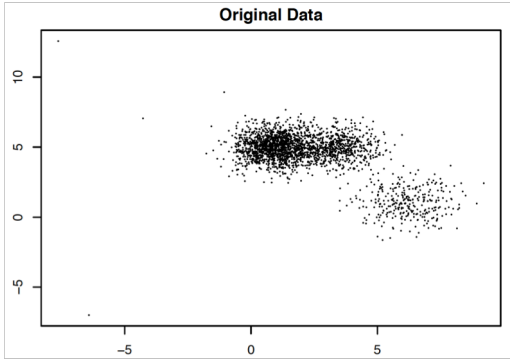
4. Sensitive to Outliers

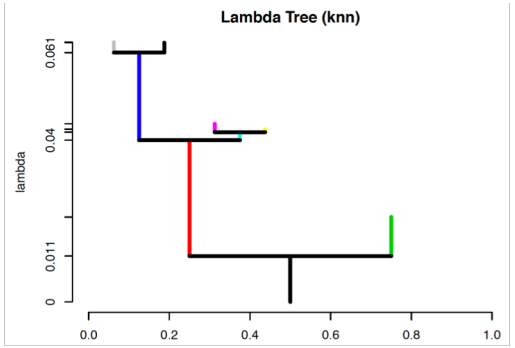


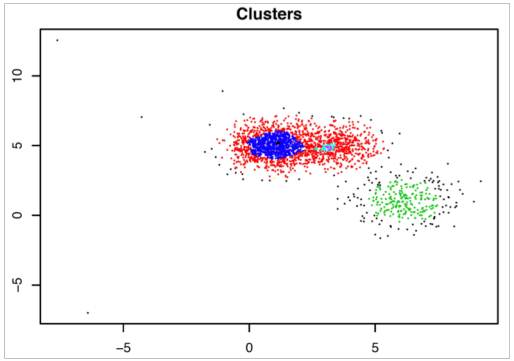
Level Set Clustering cont.

Problems:

1. Fails when clusters overlap







2. Suffers from curse of dimensionality

Summary

- All clustering algorithms come with their advantages and disadvantages.

Outline

- Introduction to Clustering Algorithms
- Motivation
- Cluster Validation
- Cluster Initialization
- Density Transformation
- Clustering Pixels in the Images
  - Application: Image Annotation

Motivation

User Dilemma:

- How do we define the pairwise similarity/distance?
- How many clusters are present in the data?
- Does the algorithm requires multiple runs?
- Does the data contain any outliers?
- Does it suffer from curse of dimensionality?
- Does the algorithm scale well with increase in datasize?
- Is it easy to tune the parameters?

Outline

- Introduction to Clustering Algorithms
- Motivation
- Cluster Validation
- Cluster Initialization
- Density Transformation
- Clustering Pixels in the Images
  - Application: Image Annotation

Determining the number of clusters

- Choose a clustering algorithm (usually K-means is used)
- Choose a cluster validation metric (Gap, BIC, AIC, Silhouette or other indices)

Determining the number of clusters

- Choose a clustering algorithm (usually K-means is used)
- Choose a cluster validation metric (Gap, Jump, Silhouette or other indices

- Run the given clustering algorithm for $k = 1, 2, \ldots, k_{max}$ (often K-means)
- For each k run the algorithm multiple times for different initialization
- Compute the validation index for each value of k and initialization
- Choose the number of clusters by maximizing the value of a given validation index.

Determining the number of clusters

- Choose a clustering algorithm (usually K-means is used)
- Choose a cluster validation metric (Gap, Jump, Silhouette or other indices

- Run the given clustering algorithm for $k = 1, 2, \dots, k_{max}$ (often K-means)
- For each k run the algorithm multiple times of better initialization
- Computer the validation index for each value of k and initialization
- Choose the number of clusters by maximizing/minimizing the value of a given validation index.

Viral Clustering (“*A Robust Method to Extract Structures in Heterogeneous Datasets*”)

- Run the algorithm once

- Start each point as a separate cluster
- Alternate between **Spread Virus** and **Suppress Virus** until convergence

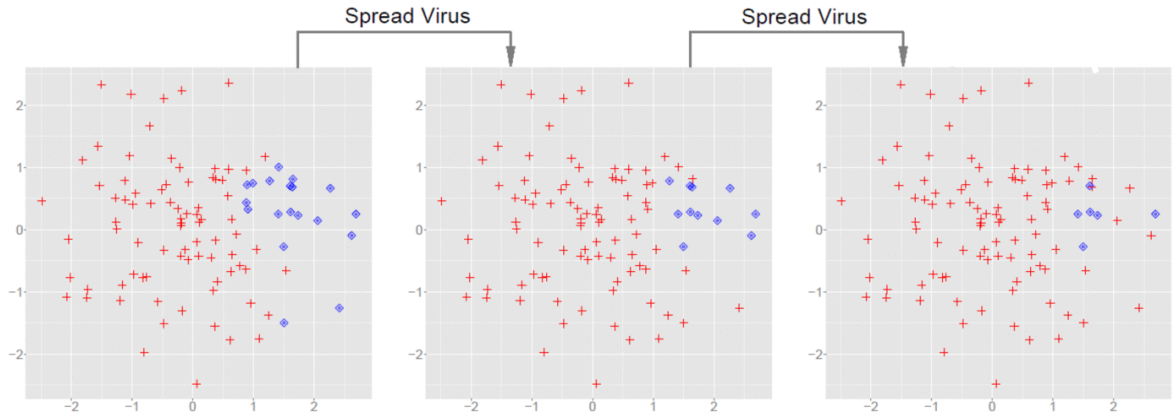
Spread Virus

1. Randomly sample a point from the smallest cluster without replacement
2. Change the label of sampled point with the label of one of its NN, randomly
3. Repeat 1 and 2, until all the points in the dataset are sampled

➢ **Interpretation**: Spread the epidemic and kill the small and well-connected clusters

Algorithm 1: Spread Virus

Input: $\mathbf{y} \in \{1, 2, \cdots, n\}^n, m$;
Initialization: $\mathbf{s} = \{1, 2, \cdots, n\}$;
while $s \neq \emptyset$ **do**
  $l \leftarrow \text{rand\_smallest}(\mathbf{y}, \mathbf{s})$;
  $\mathbf{s} \leftarrow \mathbf{s} \backslash \{l\}$;
  $l' \leftarrow \text{rand\_}m(\mathbf{D}_l)$;
  $y_l \leftarrow y_{l'}$;
end
Output: $\mathbf{y}$;



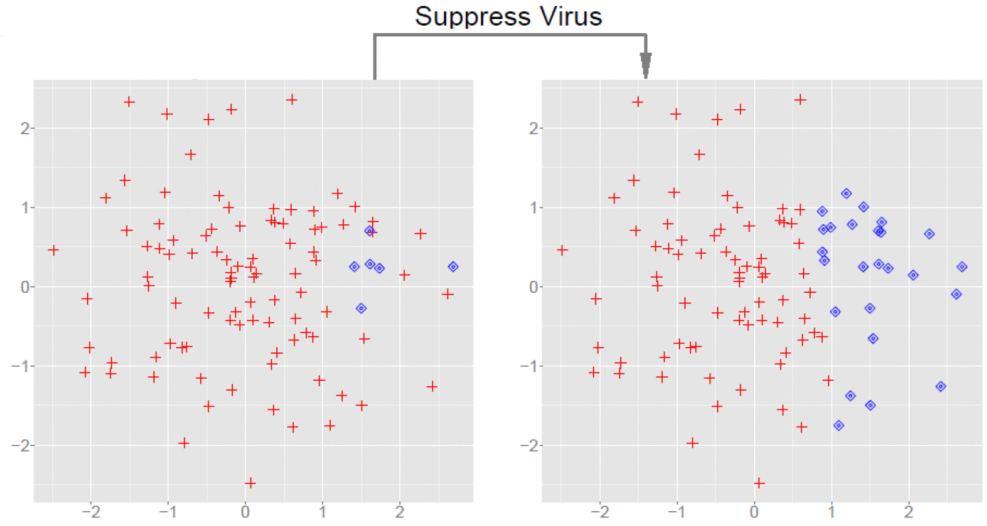
Suppress Virus

1. perform one iteration of K-means

➢ **Interpretation**: Stop the spreading epidemic based on the cluster center distance

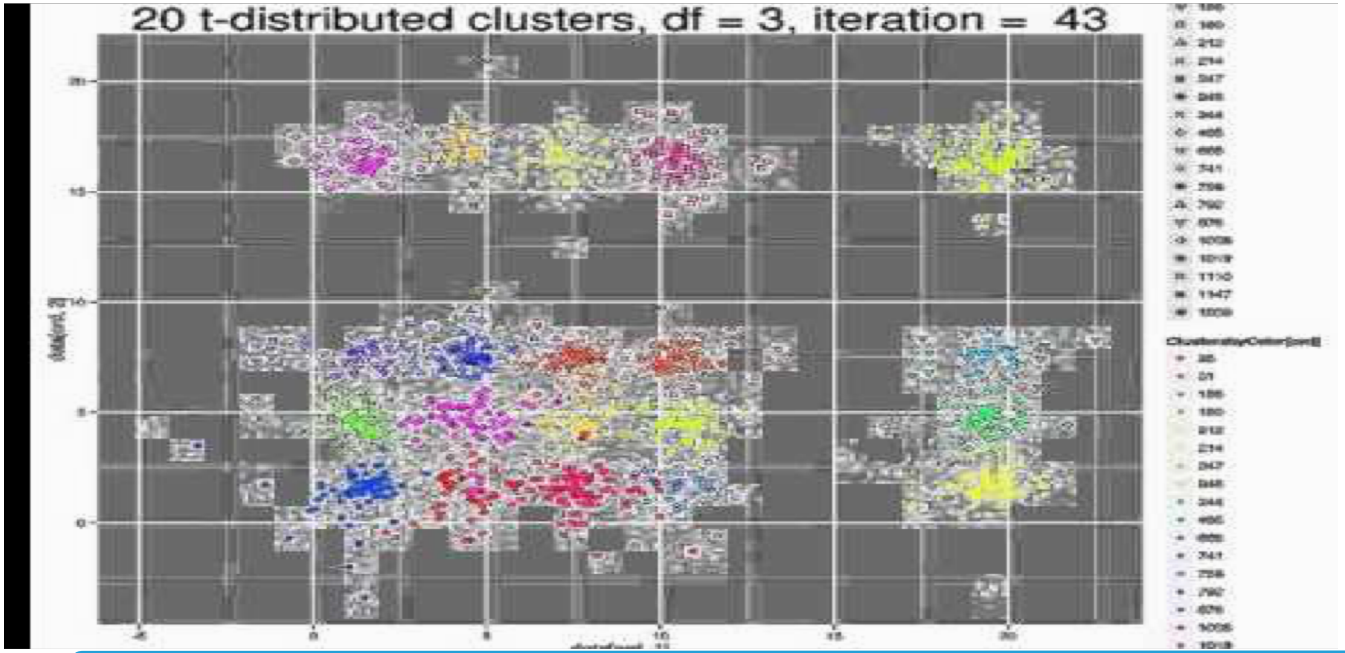
Algorithm 2: Suppress Virus

Inputs: $\mathbf{y} \in \{1, 2, \cdots, n\}^n$;
$k \leftarrow \text{nb\_clusters}(\mathbf{y})$, $\mathbf{y} \leftarrow \text{re\_index}(\mathbf{y})$;
$(\mu_1, \ldots, \mu_k) \leftarrow \text{centers}(\mathbf{y})$;
for $l = 1 : n$ **do**
  $y_l \leftarrow \arg\min_{j \in \{1,\ldots,k\}} d(\mathbf{x}_l, \mu_j)$;
end
Output: $\mathbf{y}$;

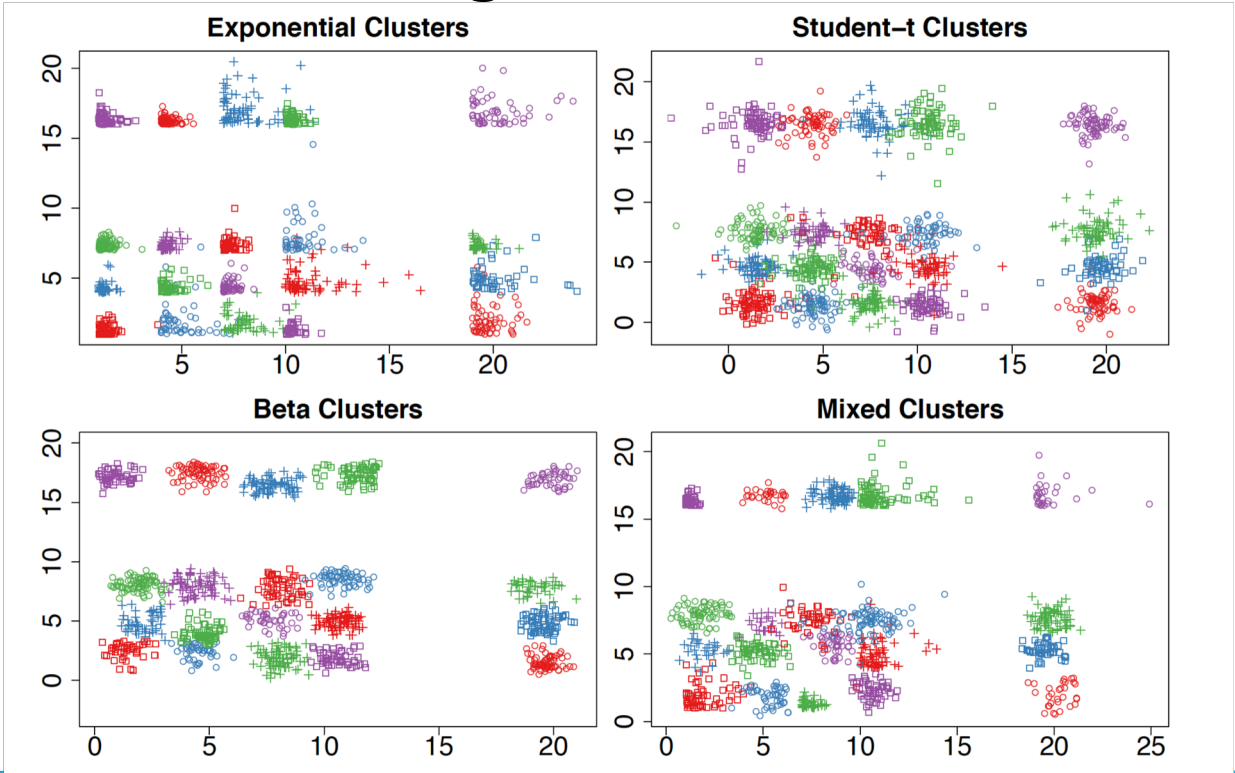


Viral Clustering: Demo

https://www.youtube.com/watch?v=h4xYrQZn5B4



Viral Clustering: Simulated Data

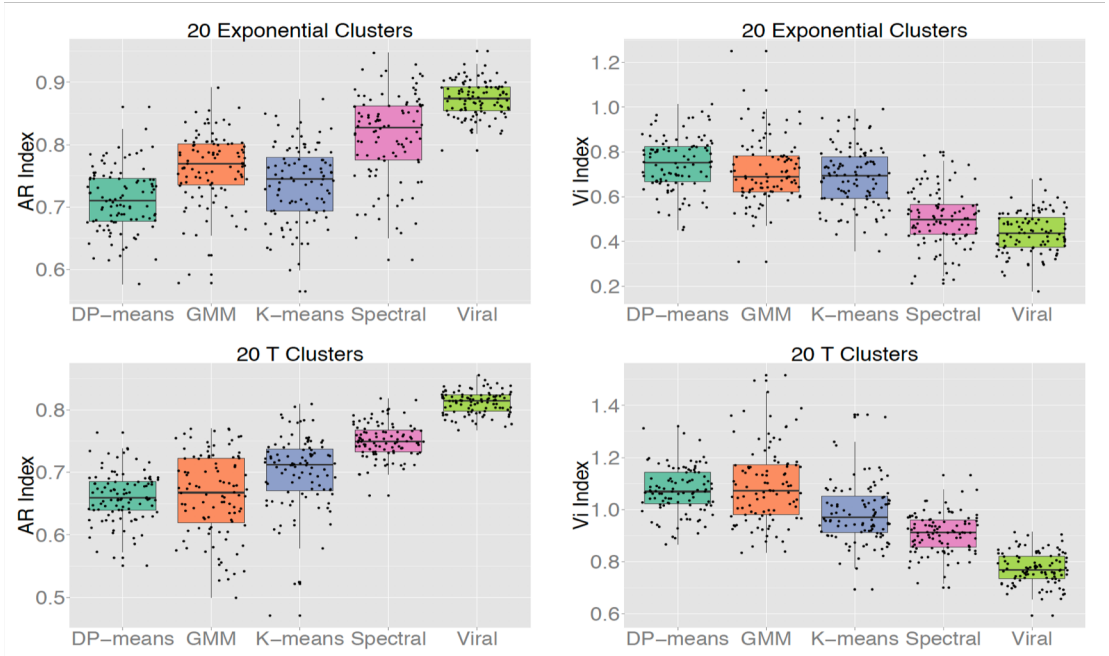


Comparison: Number of Clusters

	Exp	T	Beta	Mixture	Gaussian
	$\mu_k \pm \sigma_k$	$\mu_k \pm \sigma_k$	$\mu_k \pm \sigma_k$	$\mu_k \pm \sigma_k$	$\mu_k \pm \sigma_k$
KL	20.9±11	16±12.6	18.9±11.8	17.8±12.6	23.7±5.2
CH	31.5±5.3	25.5±4.5	24.9±3.8	26.2±6.3	24±2.5
HR	4.9±5.2	3.2±0.6	3.9±2.5	5.5±5.9	3.6±1.5
SL	19.2±7.4	3.6±2.3	5.7±4.5	9.9±5.8	2±0
Gap	2.1±1.5	1.6±0.9	1.9±1.1	1.9±1.2	2.4±1.7
BIC	23.3±1.5	22.4±5.7	21.3±2.3	20.6±2.6	1±0
Jump	26.4±5.6	21.6±1.9	20.3±4.2	23.7±6.6	39.2±0.9
VC	20±0	19.9±0.1	19.7±0.6	19.7±0.7	20±0

Fig. Mean and Standard deviation of predicted number of clusters out of 50 experiment

Comparison: Accuracy and Initialization

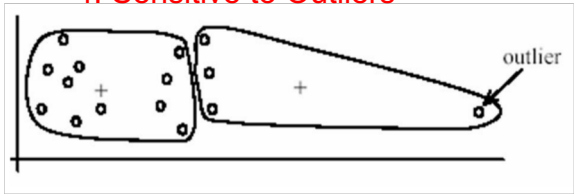


Accuracy of the various clustering algorithms on the artificial datasets (100 realizations for each dataset). **Left column:** Adjusted Rand Index (the higher, the better) and **right column:** Variation of Information Index (the lower, the better).

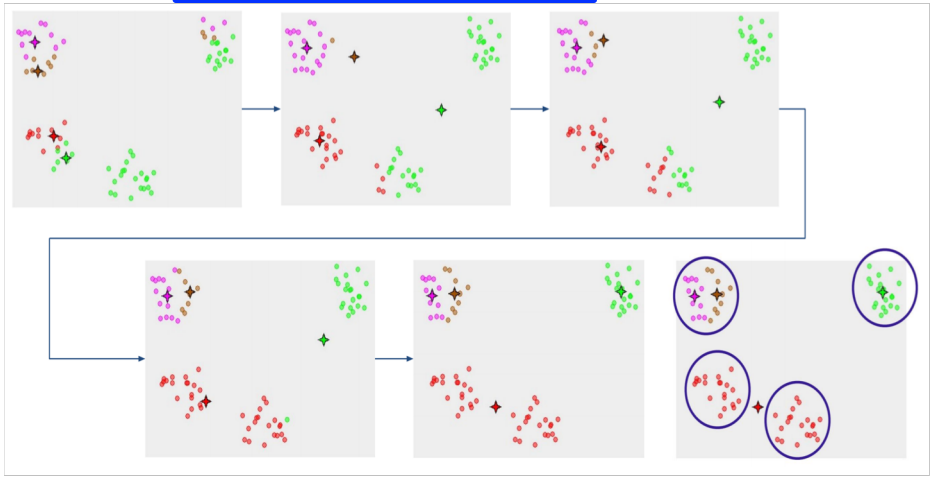
Summary

- We proposed a technique towards solving 2 major drawbacks of K-means clustering
  - Number of Clusters, k
  - Cluster Initialization
  - Distance Metric
- Drawbacks

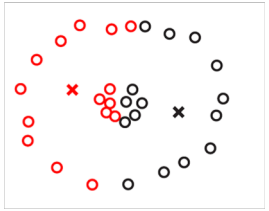
4. Sensitive to Outliers



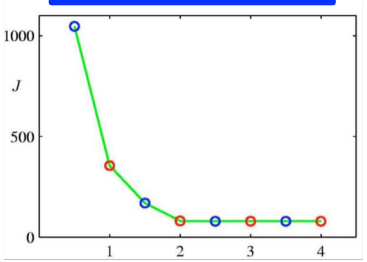
3. Sensitive to Initialization



1. Finds only center based clusters

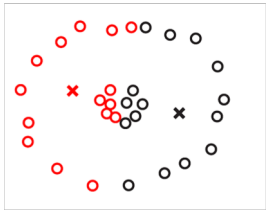


2. k is not known

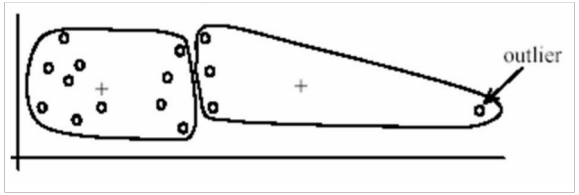


What about the following 2 cases?

Center based clusters



Sensitivity to outliers

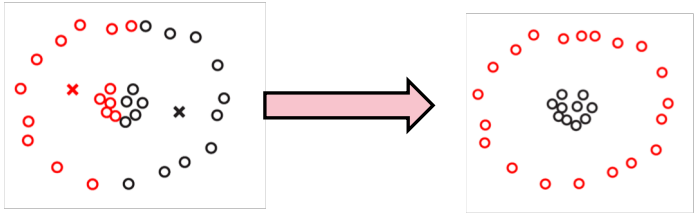


Cannot be addressed with K-means algorithm, because of its objective

Viral Spectral Clustering

Spectral Clustering

1. Input dataset X and select the number of clusters k.
2. Construct S
3. Compute normalized Laplacian L
4. Compute the first k eigenvectors $u_1, \cdots, u_k$ of L
5. Let $U \in \mathbb{R}^{n \times k}$ be the matrix containing the vectors $u_1, \cdots, u_k$ as columns
6. Form the matrix $T \in \mathbb{R}^{n \times k}$ from U by normalizing the rows to norm 1
7. Use the matrix T as an input to k-means algorithm
8. Set output of k-means as the cluster assignment of input data X



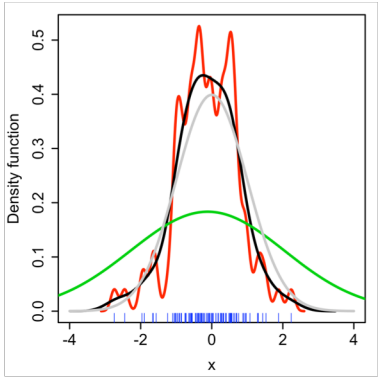
Replace K-means with Viral Clustering

More details are discuss in Chapter 4

Outline

- Introduction to Clustering Algorithms
- Motivation
- Cluster Validation
- Cluster Initialization
- Density Transformation
- Clustering Pixels in the Images
  - Application: Image Annotation

Density Estimation: Preliminaries



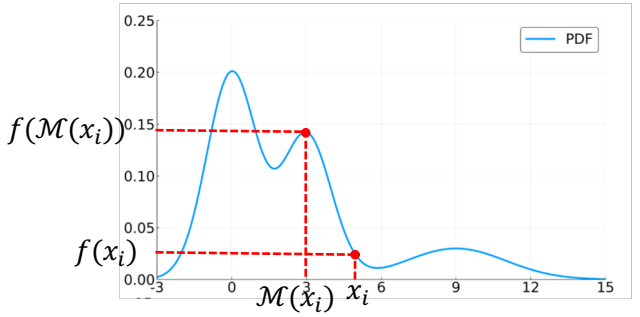
- Given: $X = \{x_1, \cdots, x_n\} \in \mathbb{R}^d$
- The k-th nearest neighbor density estimate $f_k(x)$ is defined

$$f_k(x) \triangleq \frac{k}{n \cdot v_d \cdot r_k(x)^d},$$

v_d is the volume of the unit sphere in $\mathcal{R}^d$, and $r_k(x)$ is the k-th nearest neighbor distance of point x

- The closest mode of point x_i

$$\mathcal{M}(x_i) = \underset{x_j \in C_i}{\operatorname{argmax}}[f_k(x_j)] = \underset{x_j \in C_i}{\operatorname{argmin}}[r_k(x_j)]$$



Transformed Density

- Define Modal Density (MDR) Ratio

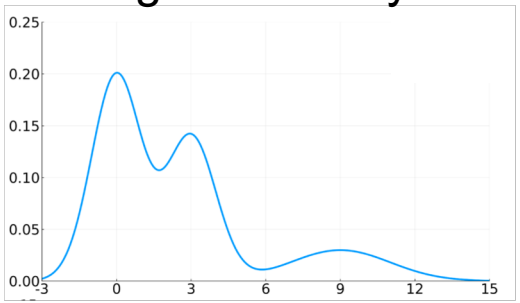
$$\theta_k(x) = \frac{r_k(x)}{r_k(\mathcal{M}(x))} = \left[\frac{f_k(\mathcal{M}(x))}{f_k(x)}\right]^{1/d}$$

- Transformed density $\tilde{f}_k(x)$

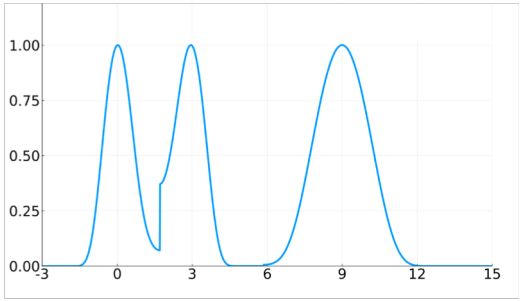
$$\tilde{f}_k(x) = e^{-\gamma[\theta_k(x)-1]}$$

Properties of Transformed Density

Original Density



Transformed Density



- Transformed density $\tilde{f}_k\ (x)$ preserves the local minima and maxima of the original density
- Transformed density $\tilde{f}_k\ (x) = 1$, for every local maximum, i.e. when $x = \mathcal{M}(x)$.
- Normalizes cluster modes to one, while increasing the gap between the clusters

Applications of $\tilde{f}_k\ (x)$: Level Set Clustering

- Level Set Clustering requires distance matrix as an input
- We modify the input of the distance matrix by the following

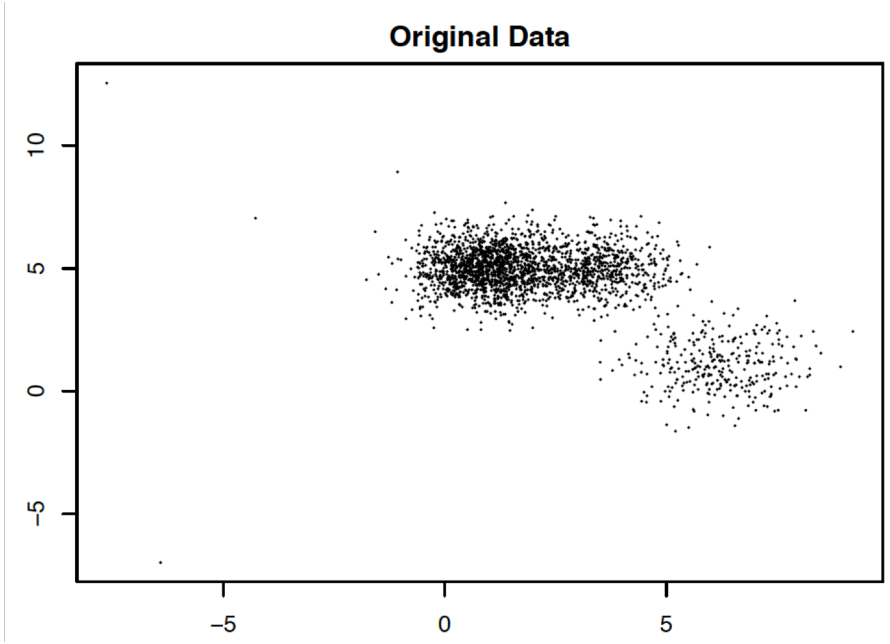
$$\tilde{d}_k(x_i, x_j) = \frac{d(x_i, x_j)}{\mathcal{K}_k(x_i, x_j)}$$

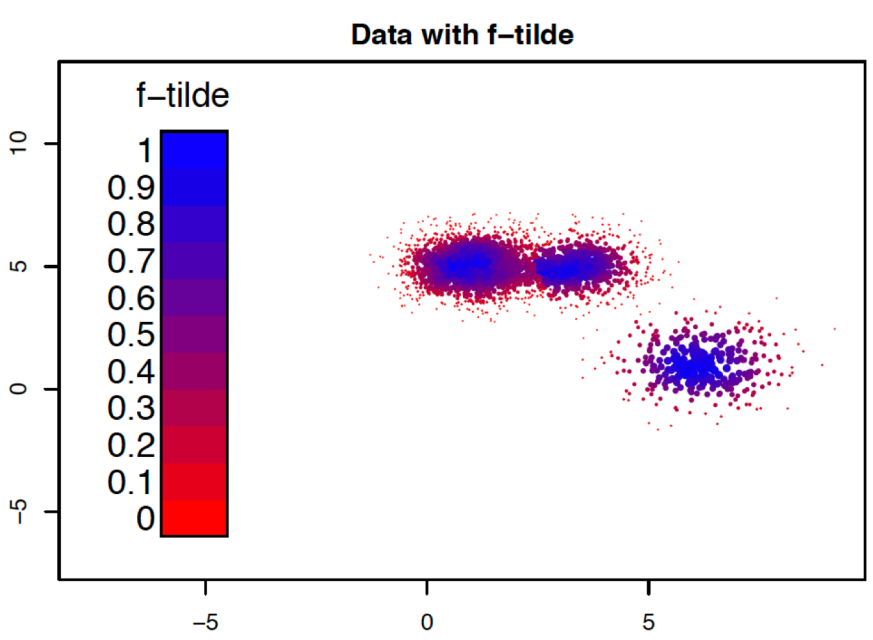
where

$$\mathcal{K}_k(x_i, x_j) = \left[\frac{\tilde{f}_k(x_i) + \tilde{f}_k(x_j)}{2}\right] \mathbb{I}\left[i \in \mathcal{N}_k(x_j)\ ||\ j \in \mathcal{N}_k(x_i)\right]$$

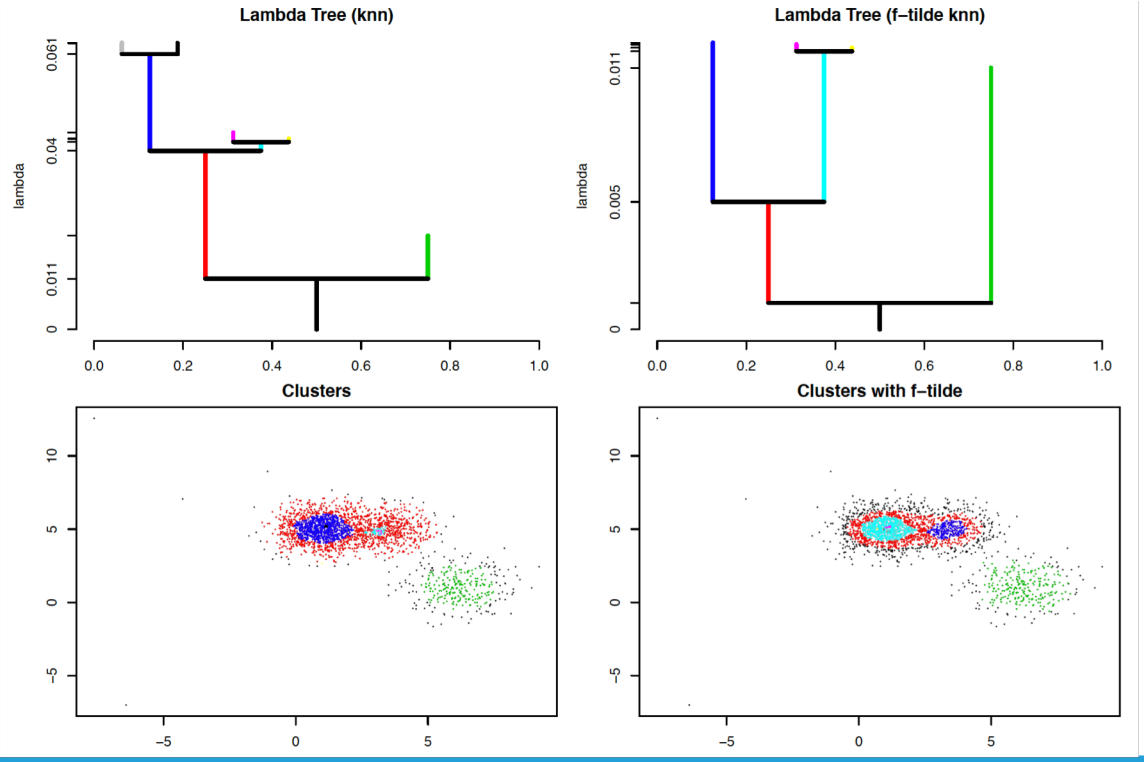
- Instead we used the modified distance as an input of level set clustering

Toy Example: $\tilde{f}_k\ (x)$ on Mixture of 3 Gaussian





Toy Example: Level-Set Clustering



Applications of $\tilde{f}_k\ (x)$: T-sne Dim. Reduction

- T-sne is a dimensionality reduction technique that takes an input the similarity matrix
- We modify the input of the similarity matrix by the following

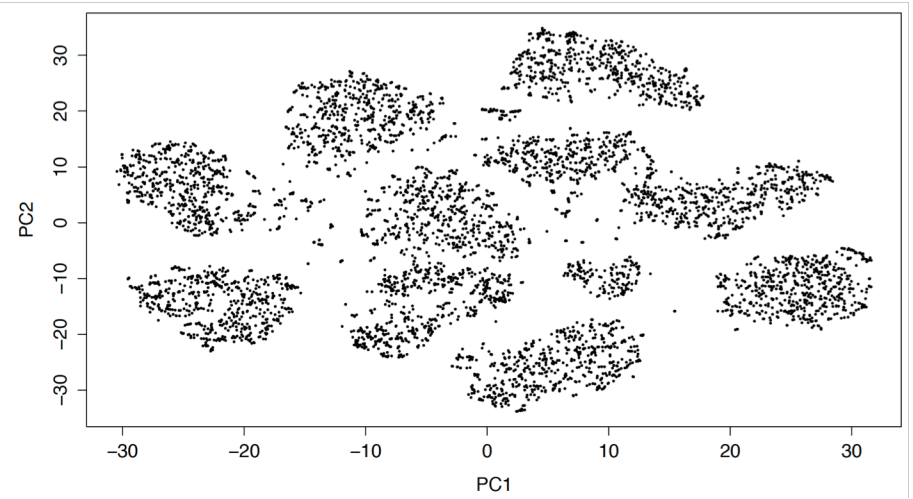
$$\tilde{S}_k(x_i, x_j) = S(x_i, x_j) \cdot \mathcal{K}_k(x_i, x_j)$$

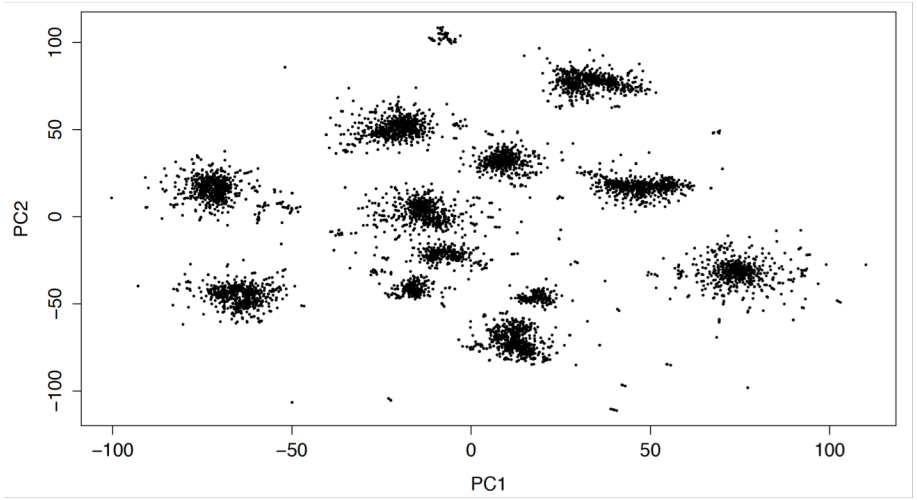
where

$$\mathcal{K}_k(x_i, x_j) = \left[\frac{\tilde{f}_k(x_i) + \tilde{f}_k(x_j)}{2}\right] \mathbb{I}\left[i \in \mathcal{N}_k(x_j) \;||\; j \in \mathcal{N}_k(x_i)\right]$$

- Instead we used the modified similarities as an input of T-sne

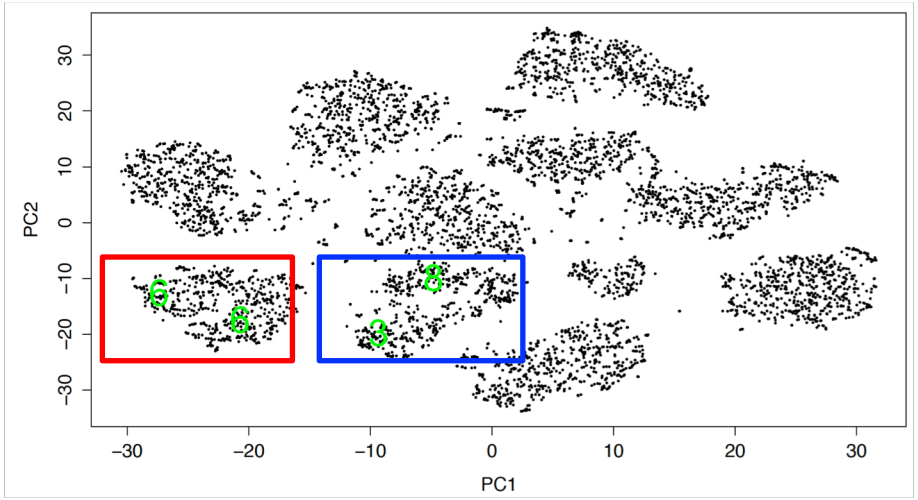
MNIST Example: T-sne

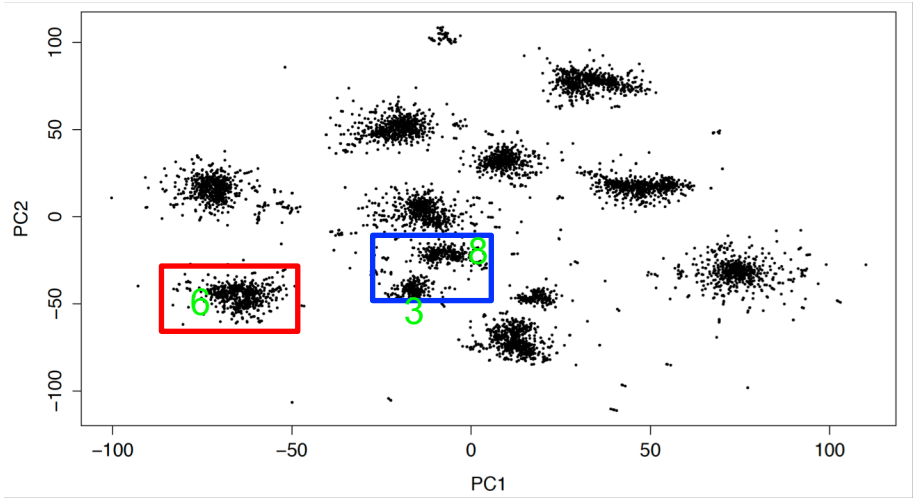




- The transformation increases the gap between the clusters
- Clusters are less uniform when transformation is applied

MNIST Example: T-sne





- The transformation increases the gap between the clusters
- Clusters are less uniform when transformation is applied

Summary

- Transformed Density, $\tilde{f}_k\ (x)$
- Defining similarity or distance using $\tilde{f}_k\ (x)$

$$\tilde{S}_k(x_i, x_j) = S(x_i, x_j) \cdot \mathcal{K}_k(x_i, x_j), \qquad \tilde{d}_k(x_i, x_j) = \frac{d(x_i, x_j)}{\mathcal{K}_k(x_i, x_j)}$$

where, $$\mathcal{K}_k(x_i, x_j) = \left[\frac{\tilde{f}_k(x_i) + \tilde{f}_k(x_j)}{2}\right] \mathbb{I}\left[i \in \mathcal{N}_k(x_j) \;||\; j \in \mathcal{N}_k(x_i)\right]$$

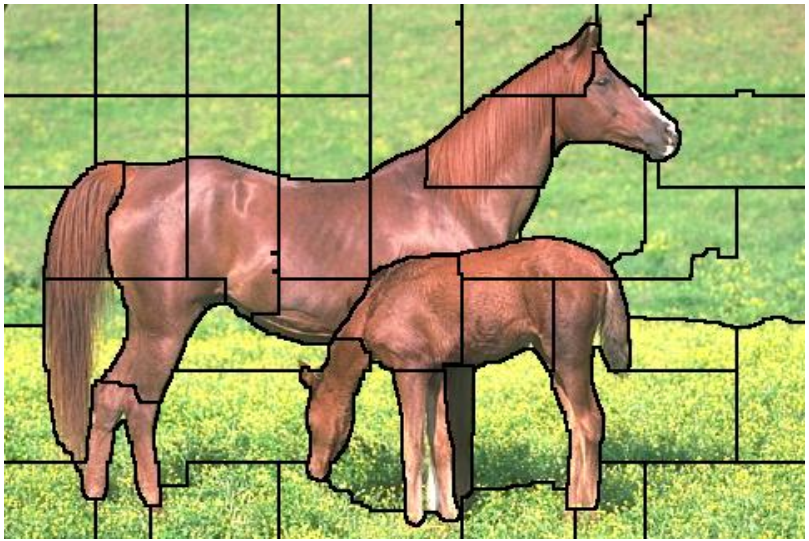
This can be viewed as semi-supervised approach for defining distance or similarity i.e. better we estimate $\tilde{f}_k\ (x)$, better the similarity or distance metric would be.

Outline

- Introduction to Clustering Algorithms
- Motivation
- Cluster Validation
- Cluster Initialization
- Density Transformation
- Clustering Pixels in the Images
  - Application: Image Annotation

Clustering Pixels in the Images

- Superpixel algorithms group neighboring pixels with similar color intensities into small patches such that object boundaries are well preserved.



Desirable properties of Superpixel Algorithms

- Boundary adherence
- Compactness
- Smoothness
- Size homogeneity
- Runtime
- Number of superpixels

Desirable properties of Superpixel Algorithms

- Boundary adherence
- Compactness
- Smoothness
- Size homogeneity
- Runtime
- Number of superpixels

Motivation: Create an algorithm that will satisfy all the requirements above

Setup

- Given
  - The image $\mathrm{X} = \{x_1, \ldots, x_n\}$
  - Image contour $C = \{c_1, \ldots, c_n\}$
  - Weighted neighborhood graph of degree 4, $G = \{\mathcal{V}, \mathcal{E}, W\}$

- Define:
  - Contour pixel similarity: $w_{l,l'} = e^{-\frac{c_l + c_{l'}}{\sigma_1}}$, if $(l, l') \in \mathcal{E}$

Cut and Average Linkage

- For the given superpixels S_i and S_j define

$$\text{Cut}(S_i, S_j) = \sum_{\ell \in S_i} \sum_{\ell' \in S_j} w_{\ell,\ell'}$$

- Define average linkage as

$$\text{Link}(S_i, S_j) = e^{-\frac{||\mu_i - \mu_j||_2}{\sigma_2}}$$

where μ_i and μ_j are average color intensity of S_i and S_j

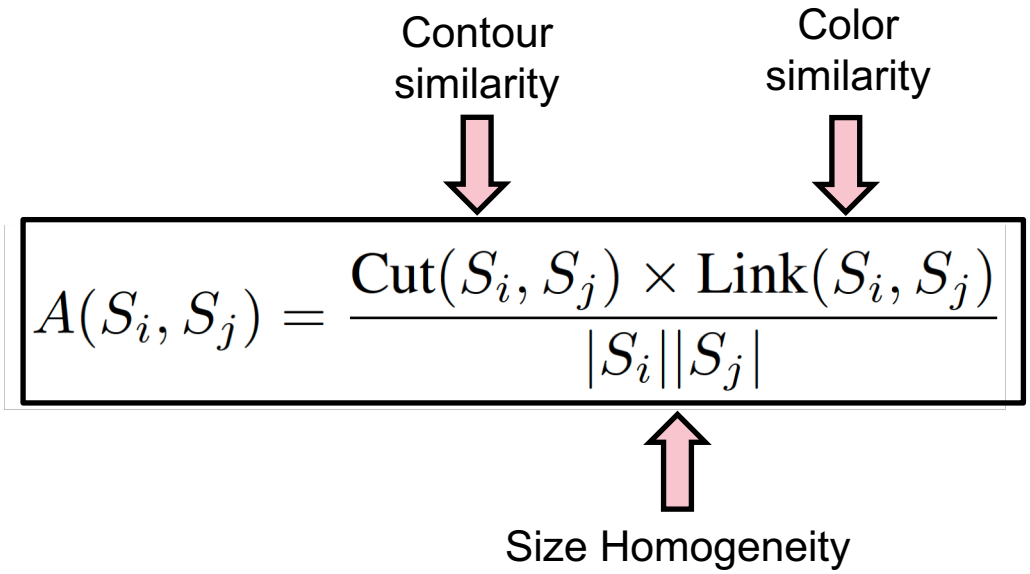
Superpixel Similarity

Define the Superpixel Similarity

$$A(S_i, S_j) = \frac{\text{Cut}(S_i, S_j) \times \text{Link}(S_i, S_j)}{|S_i||S_j|}$$

Superpixel Similarity

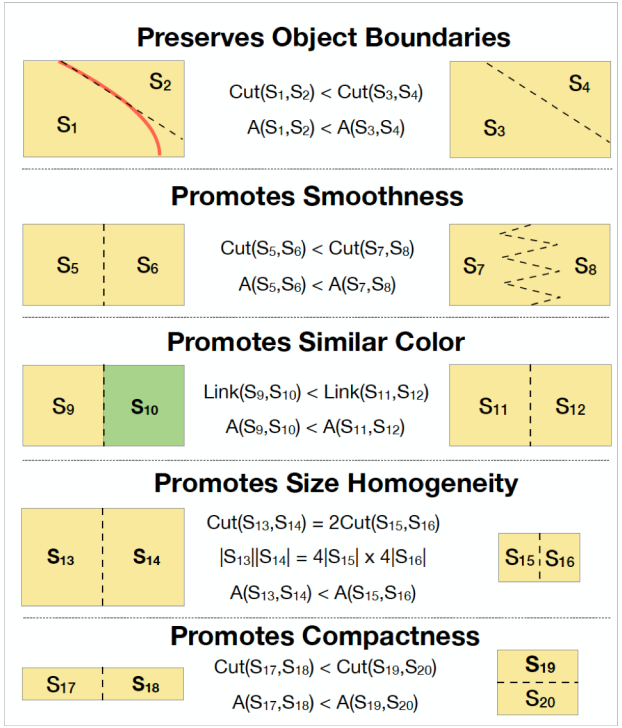
Define the Superpixel Similarity



Choice of the Superpixel Similarity

- Algorithm: greedy hierarchical merge, which promotes the characteristics we discussed earlier.

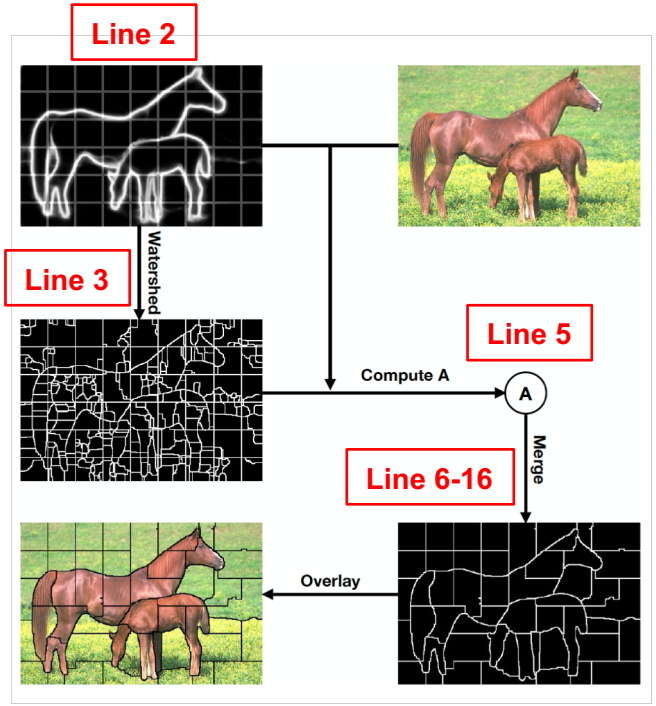
$$A(S_i, S_j) = \frac{\mathrm{Cut}(S_i, S_j) \times \mathrm{Link}(S_i, S_j)}{|S_i||S_j|}$$



PALC Superpixels

Algorithm 7: PALC Superpixels

Input: image $\mathcal{X}$, contour $C, k, \sigma_1, \sigma_2, h$;
$C_G = C + \text{Grid}(\mathcal{X}, k, \sigma_1)$;
$\mathcal{S} \leftarrow \text{Watershed}(C_G)$;
$K \leftarrow \max(\mathcal{S})$;
Compute $A = (A(S_i, S_j))_{i,j \in \{1,\dots,K\}}$ from $\mathcal{X}$, C_G and $\mathcal{S}$ using Eq. (6.1) ;
while $K > k$ **do**
  Choose $(i, j) \in \text{argmax}\, A(S_i, S_j)$;
  if $\frac{|S_i|+|S_j|}{|S_{(1)}|} < h$ **then**
    $S_i \leftarrow S_i \cup S_j, \mathcal{S} \leftarrow \mathcal{S} \setminus \{S_j\}$
  else
    $i' \in \text{argmax}\, A(S_{(1)}, S_{i'})$;
    $S_{i'} \leftarrow S_{(1)} \cup S_{i'}, \mathcal{S} \leftarrow \mathcal{S} \setminus \{S_{(1)}\}$
  end
  Update A;
  $K \leftarrow K - 1$;
end
Output: $\mathcal{S}$;



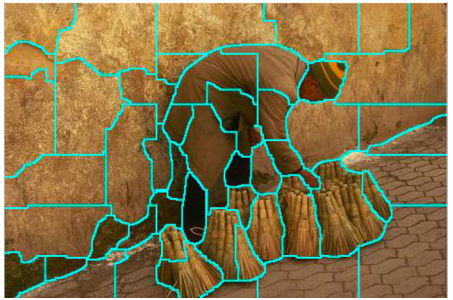
Effect of parameter h

Algorithm 7: PALC Superpixels

Input: image $\mathcal{X}$, contour C, k, σ_1, σ_2, h;
$C_G = C + \text{Grid}(\mathcal{X}, k, \sigma_1)$;
$\mathcal{S} \leftarrow \text{Watershed}(C_G)$;
$K \leftarrow \max(\mathcal{S})$;
Compute $A = (A(S_i, S_j))_{i,j \in \{1,\dots,K\}}$ from $\mathcal{X}$, C_G and $\mathcal{S}$ using Eq. (6.1) ;
while $K > k$ **do**
  Choose $(i, j) \in \text{argmax } A(S_i, S_j)$;
  if $\frac{|S_i|+|S_j|}{|S_{(1)}|} < h$ **then**
    $S_i \leftarrow S_i \cup S_j, \mathcal{S} \leftarrow \mathcal{S} \setminus \{S_j\}$
  else
    $i' \in \text{argmax } A(S_{(1)}, S_{i'})$;
    $S_{i'} \leftarrow S_{(1)} \cup S_{i'}, \mathcal{S} \leftarrow \mathcal{S} \setminus \{S_{(1)}\}$
  end
  Update A;
  $K \leftarrow K - 1$;
end
Output: $\mathcal{S}$;

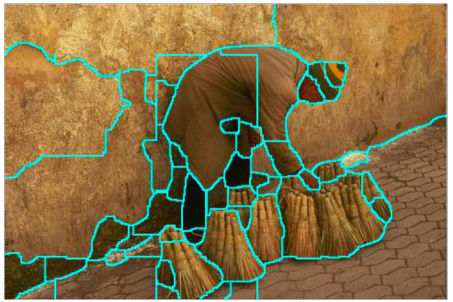
Original Image

PALC with h=10

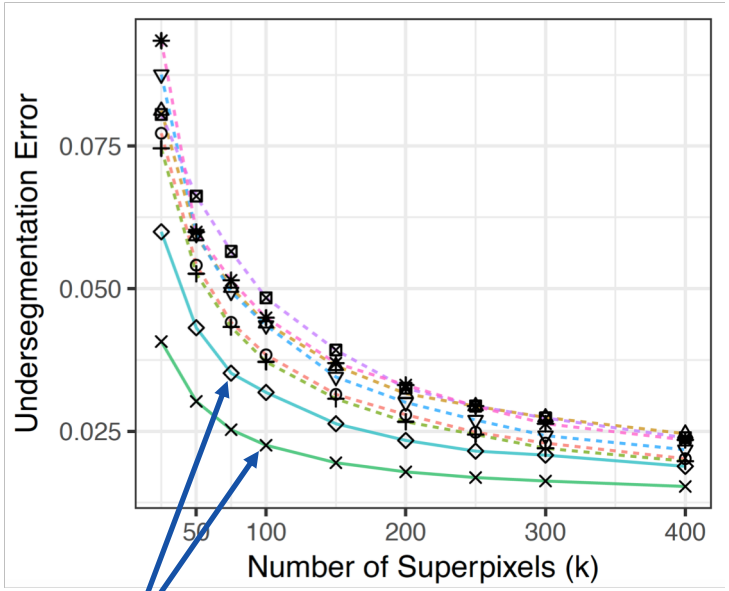


PALC with h=30

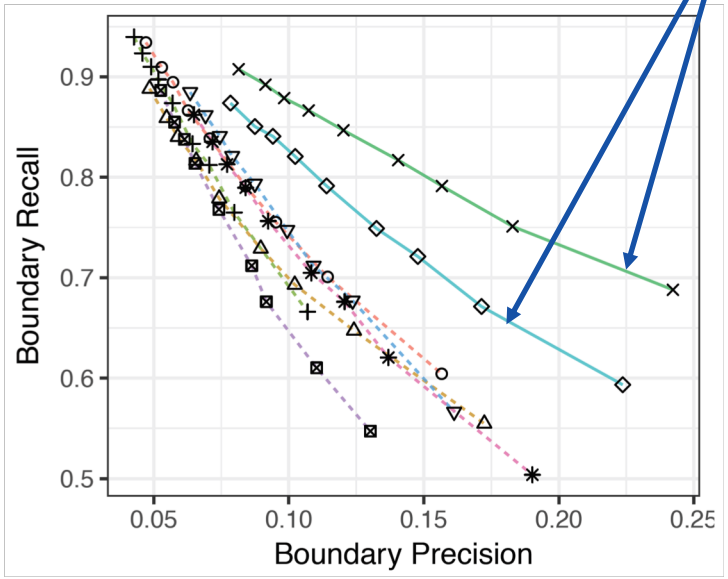
PALC with h=100



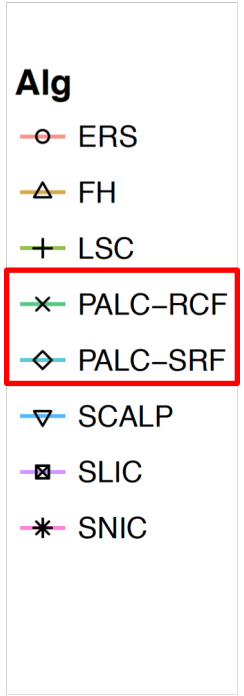
Experiments



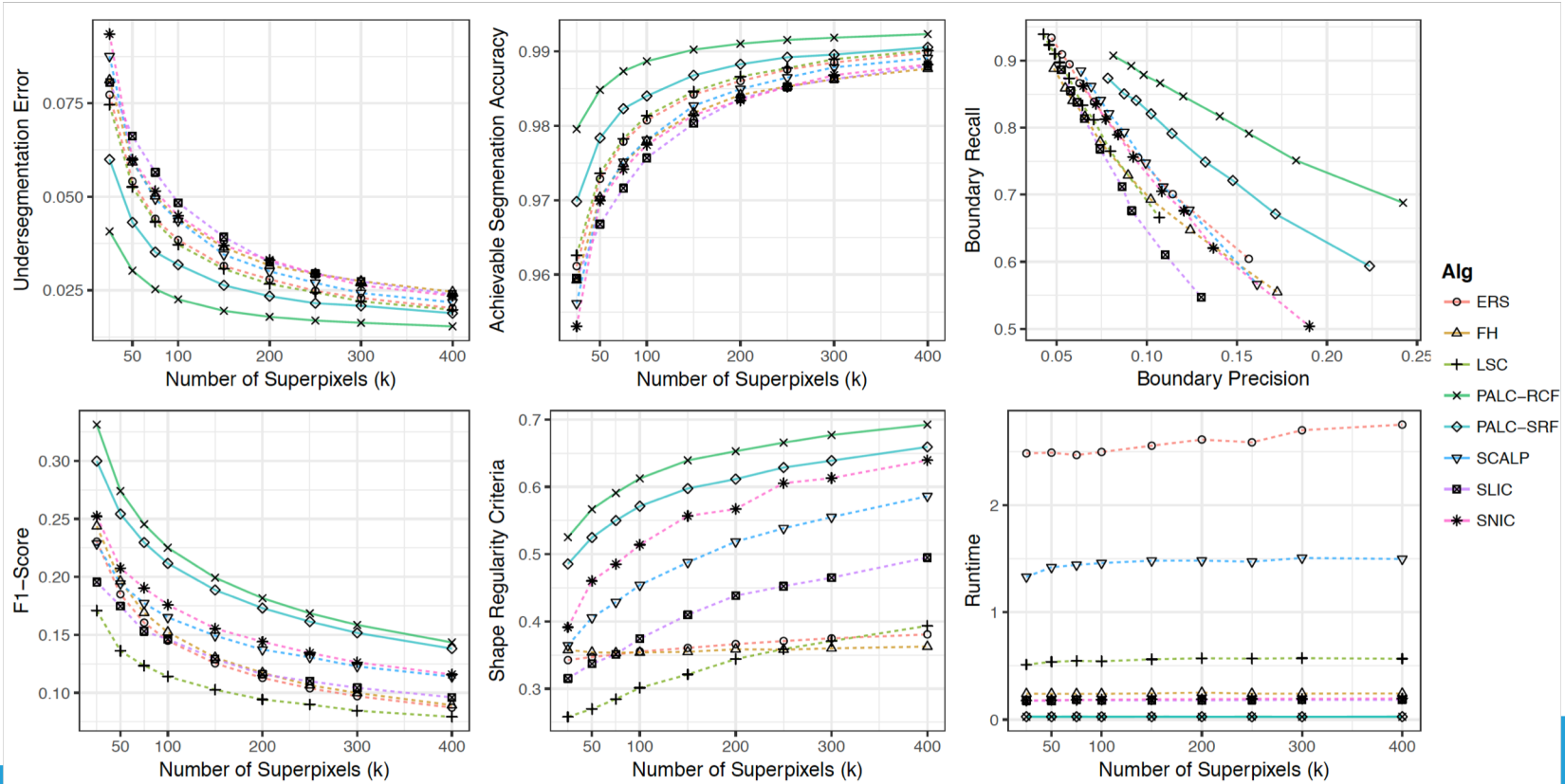
Contour Matters!



Contour Matters!



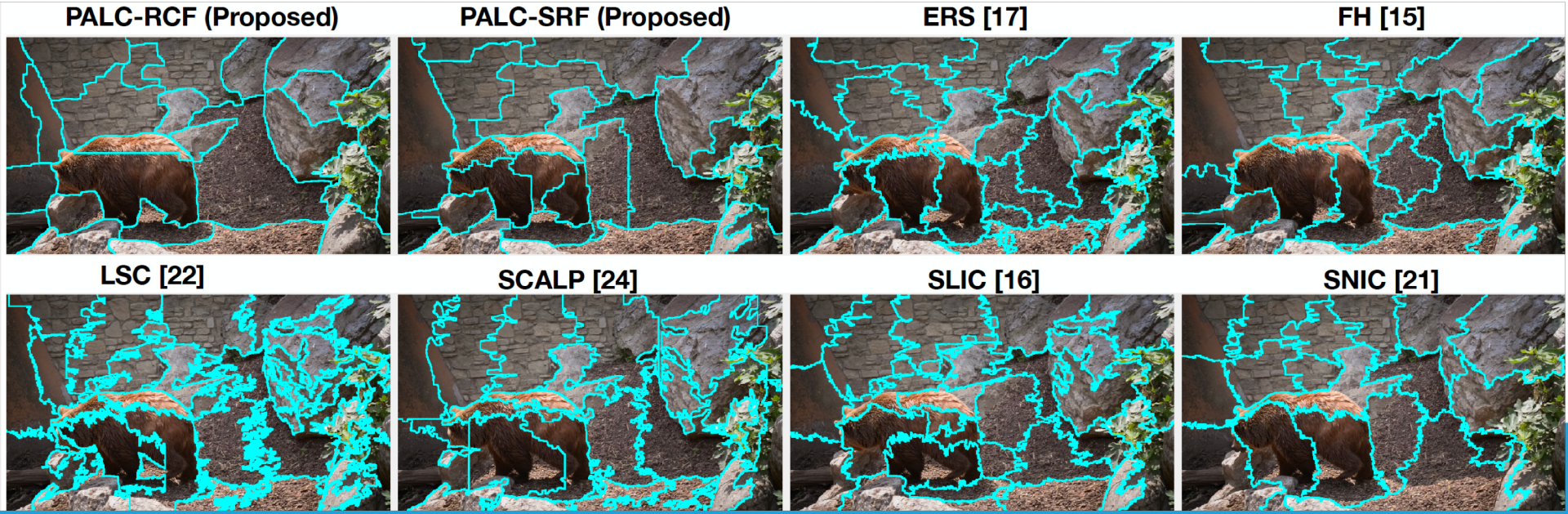
Experiments on DAVIS dataset



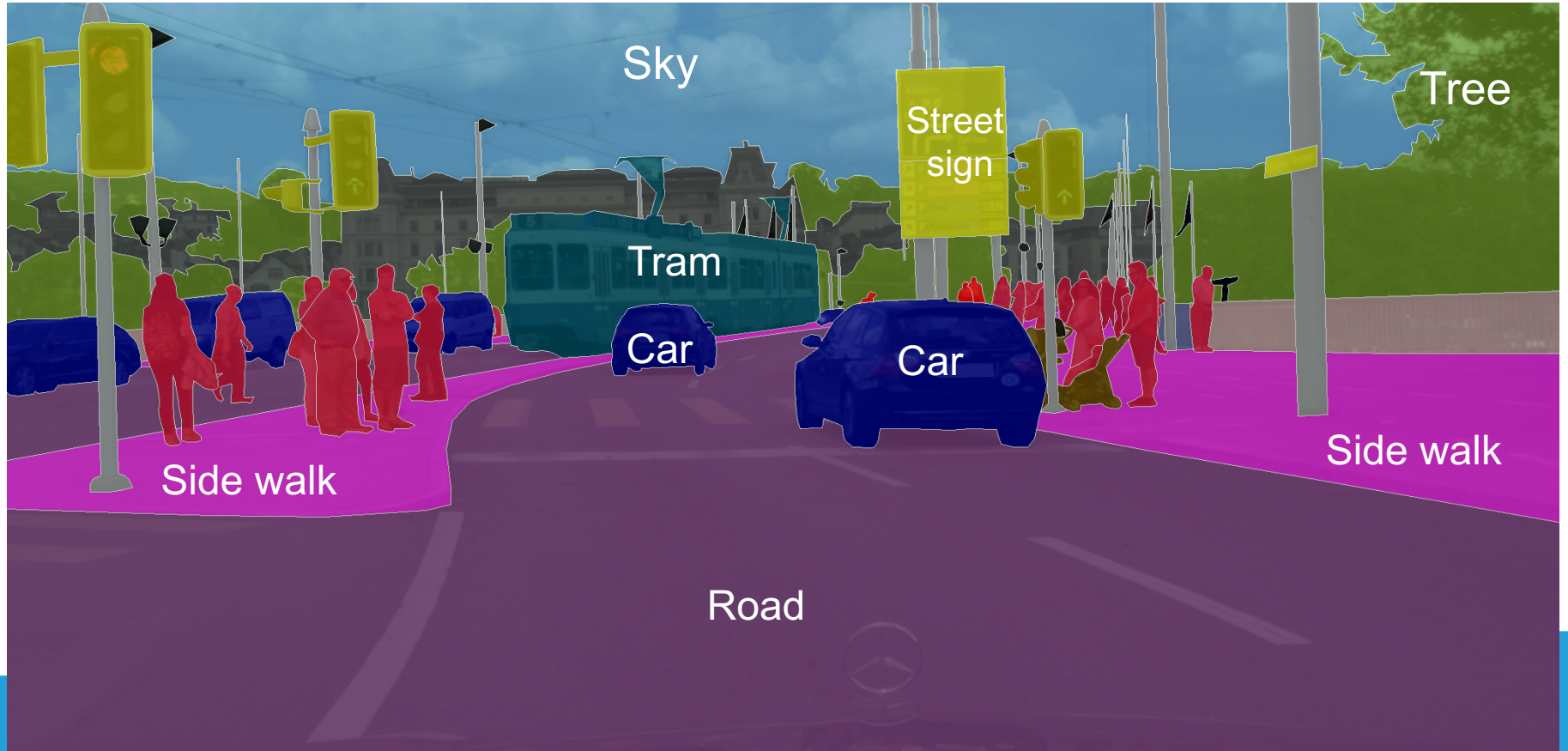
Visual Comparison

PALC

- Preserves object boundaries
- Has smooth boundaries
- Has fast runtime
- Promotes Similar Color
- Promotes Size Homogeneity
- Promotes Compactness



Applications: Image Annotation



Applications: Image Annotation

- Deep learning based computer vision algorithms are data hungry
- Data quality often is more important than state-of-the-art algorithm
- Hence, image annotation software became a hot research topic in the computer vision community

Applications: Image Annotation

- Deep learning based computer vision algorithms are data hungry
- Data quality often is more important than state-of-the-art algorithm
- Hence, image annotation software became a hot research topic in the computer vision community

Image Annotation Software based on PALC Superpixels!

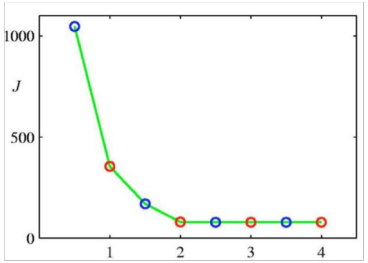
Advantages of PALC Annotation

- PALC runs **10x faster** than some of the fastest segmentation-based technologies used in annotation.
- PALC is **extremely accurate** and **generates non-homogeneous regions**. This allows to select both large and small objects with just one click.
- PALC allows to **change** the number of segments **instantly.**
- PALC has a **manual correction feature** for the regions that need further editing.
- PALC also has a **self-learning feature** (i.e. the segmentation quality improves with more and more annotated data).

Conclusion

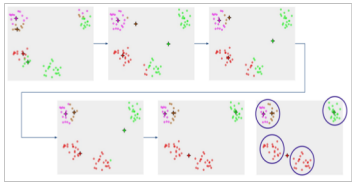
Chapter 3

Validation



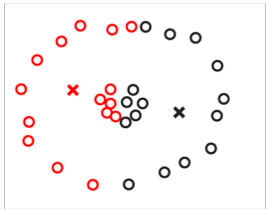
Chapter 3,4

Initialization



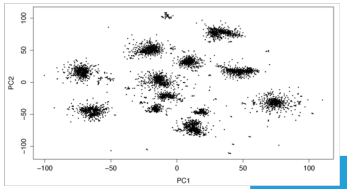
Chapter 4

Finds only center based clusters



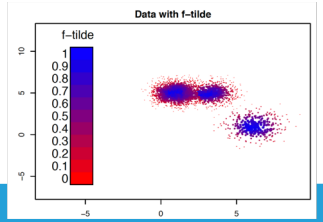
Chapter 5

Dimensionality reduction



Chapter 5

Density estimation and transformation



Chapter 6

Clustering pixels in the images

Chapter 7

Applications

Future Work

1. **Clustering**: combine various components in Chapter 3-6 towards designing an algorithm with the following features:
    - Probabilistic outcome
    - Scale easily to millions of datapoints
    - Not sensitive to curse of dimensionality
    - Not sensitive to initialization
    - Not sensitive to outliers
    - Automatically adjust the number of clusters
    - Have easy interpretable and tunable parameters

Developed the code and already tested. Works like a charm ☺

Future Work cont.

Video Annotation: combine various components in Chapter 6-7 towards designing a pixel accurate video annotation software

Thank you for your attention.