# Multimedia Technology

## Lecture 5: Unsupervised Learning

Lecturer: *Dr*. Wan-Lei Zhao

*Autumn Semester* 2024

Email: wlzhao@xmu.edu.cn, 

# Outline

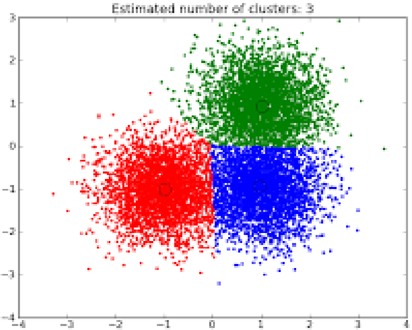
# Topics in this Lecture

- We are going to leave apart from IR for a while
- We are going to introduce several extremely useful machine learning algorithms
  - While you can say they are data-mining tools/algorithms
- Not all machine learning algorithms will be discussed
  - Only the popular algorithms will be covered
  - We are going to use them in the lectures coming next
- Why I do so
  - I try to make this course self-sufficient and self-containing
  - Considering that you come from different places with different backgrounds

# Outline

# General Concept about clustering (1)

- Given a dataset (with N number of items)
- Clustering make a partition on the dataset
- Data items have been divided into **k** groups
- **k** is usually given by user



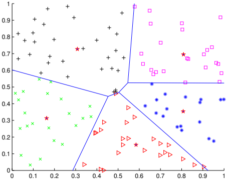
# General Concept about clustering (2)

- Clustering is a hot research topic in 1990s in the heyday of data-mining
- There are more than 10 different clustering algorithms in the literature
- They have been built upon different assumptions in different contexts
  - **k-means**: general purpose, K is required as input parameter
  - **DBSCAN** and **mean-shift**: density based approach, distance threshold or density threshold is required
  - **Chameleon** and **Agglomerative Approach**: down-to-top approach
  - **Normalize cut**: proposed under the context of image segmentation

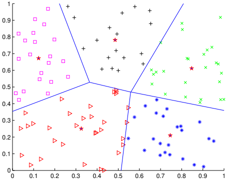
# k-means: the general procedure

- It is a chicken-egg loop
- Given **N** items and **K**
  1. Select K items out as initial centers
     1. Assign items to its closest center (a partition is formed)
     2. Update each center with average (or centroid) of items in this group
  2. Loop until centers do not change
- The complexity is $O(K \cdot N \cdot D)$, where **D** is the dimension of data item
- This is the most efficient clustering, and it can be faster!!
- Only one parameter
- It converges quickly
- Dark side1: **Be careful** if **K** is a critical number in your application
- Dark side2: it only obtains sub-optimal solution, this is true for all clustering algorithms

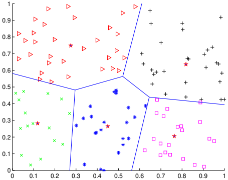
# k-means: a demo



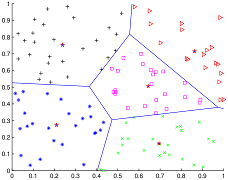

(a) iter=0



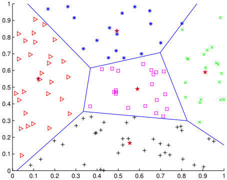

(b) iter=1



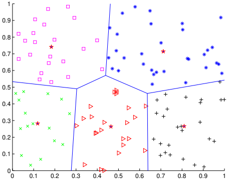

(c) iter=2




(d) iter=3

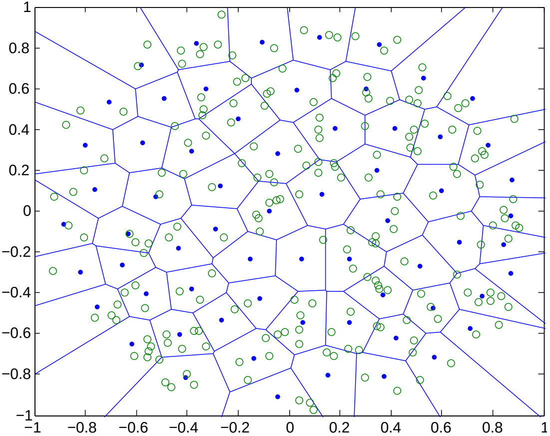



(e) iter=4




(f) iter=5

# k-means: additional advantage




- k-means forms a convex partition on the whole space
- Known as Voronoi cells
- Each cell is scoped by one cluster center

# Variants of k-means: k-means++ (1)

- This work is still quite new[1]
- Motivation: try to optimize the initialization of clustering centers
- Idea: try to select points far apart from each other
- Goal: adapt better to the data distribution

[1] D. Arthur and S. Vassilvitskii, "k-means++: the advantages of careful seeding", 18th ACM-SIAM symposium on Discrete algorithms, 2007.

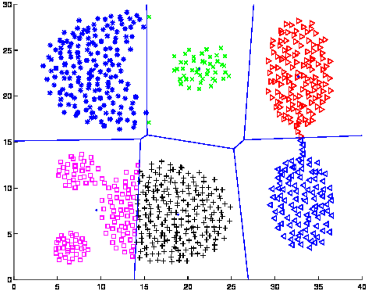
# Variants of k-means: k-means++ (2)

- Given **N** items and **K**
  1. Select one item out randomly as the first center
  2. Repeat following procedure **K-1** times
     1. Calculate distance for each item **x** to existing center(s)
     2. Take the distance that each item to its cloest center as D(**x**)
     3. Select a new center out with probability propotional to $D^2(x)$
     4. Join this new center to existing centers
  3. Complete k-means clustering according to conventional procedure
- Modifications are made only on the initialization stage
- This leads to faster convergence
- Better adaptation to the data distribution

# Outline

# Motivation: $k$-means remains pupolar (1)

- $k$-means is ranked at top-10 algorithms in data-mining
- It remains popular in various applications
  - Large-scale web page clustering
  - Large-scale Image clustering/linking
  - Vector quantization and product quantization
  - Data Compression



# Motivation: superity of $k$-means (2)

- Advantages
  - Simple
  - Fast, the complexity is $O(n{\cdot}d{\cdot}k{\cdot}t)$
    - **n** is the size of data
    - **d** is the dimension of the data
    - **k** is the number of clusters
    - **t** is the number of iterations
- *Comments:*
  - Compared to mean-shift, DB-SCAN, etc.
  - It is much more efficient
  - In terms of clustering quality
  - The results are moderately good in most of the cases

# Motivation: disadvantage of $k$-means (3)

- Disadvantages
  - It is **slow**, the complexity is $O(n \cdot d \cdot k \cdot t)$
    - **n** is the size of data
    - **d** is the dimension of the data
    - **k** is the number of clusters
    - **t** is the number of iterations
- *Comments:*
  - Given **n** is big
  - Given **d** is high
  - Given **k** is large
  - Given **t** is large too!
- *For instance:*
  - $2M \times 128$ matrix
  - Divide into *20,000* clusters
  - Run on 3.4G Hz, 4 threads
  - It takes more than **3** days

# Motivation: could be faster and better? (4)

- It is **slow**, the complexity is $O(n \cdot d \cdot k \cdot t)$
  - **n** is the size of data
  - **d** is the dimension of the data
  - **k** is the number of clusters
  - **t** is the number of iterations
- *Possible solutions:*
  - We cannot change **n**
  - We cannot change **d**
  - We can reduce **k** to log(**k**) by hierarchical clustering
  - We can make **t** smaller, that means it converges faster

# Traditional $k$-means: a recap

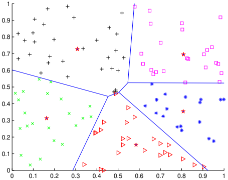
1. Initialize **k** centroids
2. Assign each sample to its closest centroids
3. Recompute centroids with assignments produced in **Step 2**
4. Repeat **Step 2** and **Step 3** until convergence

$k$-means is formulized as a minimization problem:

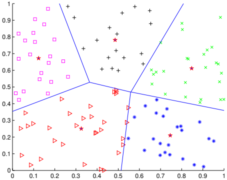
$$\text{Min.} \sum_{q(x_i)=r} \| C_r - x_i \|^2. \tag{1}$$

where $q(x_i)$ returns the closest centroid $C_r$ for $x_i$.

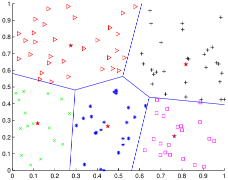
# *k*-means: a demo



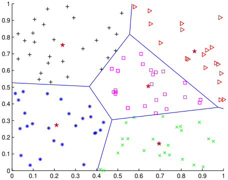
(a) iter=1



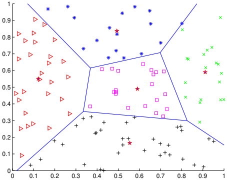
(b) iter=2



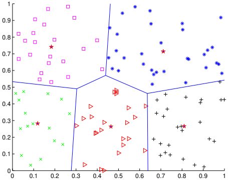
(c) iter=3



(d) iter=4



(e) iter=5



(f) iter=6

# Formalize $k$-means in a new form

- Given $D_r = \sum_{x_i \in S_r} x_i$ and $n_r = |S_r|$
- $C_r$ is the centroid of cluster $S_r$

$$\text{Min.} \sum_{q(x_i)=r} \| C_r - x_i \|^2 \tag{1}$$

$$\Downarrow$$

$$\text{Max. } \mathcal{I}_1 = \sum_{r=1}^{k} \frac{D_r' D_r}{n_r} \tag{2}$$

- It takes a little bit of efforts to work out above result
- We are happy to see this result (later you will see)

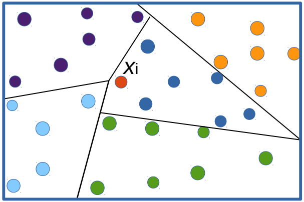
## Optimize $k$-means with new target function

- Given $D_r = \sum_{x_i \in S_r} x_i$ and $n_r = |S_r|$
- $C_r$ is the centroid of cluster $S_r$

$$\text{Max. } \mathcal{I}_1 = \sum_{r=1}^{k} \frac{D_r' D_r}{n_r} \tag{2}$$

- Now we have new optimization function
- Problem: how to maximize $\mathcal{I}_1$?

# Optimize $k$-means with new target function




1. Pick $x_i$ in random ($x_i \in S_u$)
2. Check $\Delta\mathcal{I}_1$ when moving $x_i$ from cluster $S_u$ to $S_v$

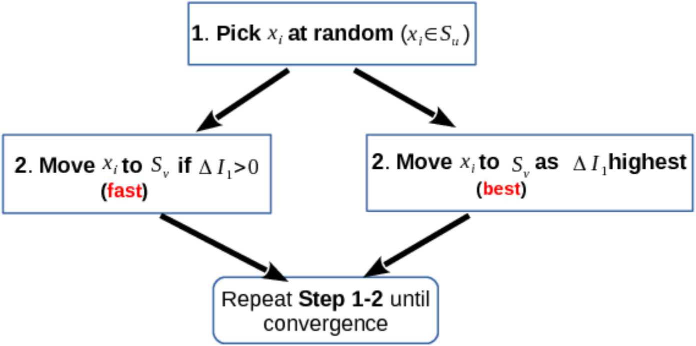
$$\begin{aligned}\Delta\mathcal{I}_1(x_i) &= \frac{(D_v + x_i)'(D_v + x_i)}{n_v + 1} - \frac{(D_u - x_i)'(D_u - x_i)}{n_u - 1} \\ &= \frac{D_v' D_v + 2x_i' D_v + x_i' x_i}{n_v + 1} - \frac{D_u' D_u - 2x_i' D_u + x_i' x_i}{n_u - 1}\end{aligned} \tag{3}$$

3. Move $x_i$ to $S_v$ that achieves highest $\Delta\mathcal{I}_1$

# Outline of the algorithm (1)

1. Assign $x_i \in X$ with a random label
2. Calc. $D_1, \cdots, D_r, \cdots, D_k$ and $\mathcal{I}_1$
3. Repeat
4. For each $x_i \in X$
5. Seek $S_v$ that max. $\Delta\mathcal{I}_1(x_i)$
6. If $\Delta\mathcal{I}_1(x_i) > 0$
7. Move $x_i$ to cluster $S_v$
8. End-If
9. End-For
10. End-Repeat

# $k$-means$^{\#}$ in breif



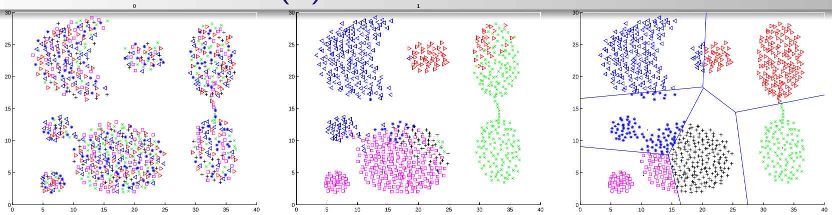

- Comments:
  - We can either choose "best" move or "fast" move
  - "fast" converges to lower distortion but takes more rounds
  - "best" converges faster but slower in each iteration

# $k$-means$^{\#}$ and $k$-means: major differences

| Operations | $k$-means$^{\#}$ | $k$-means |
|---|---|---|
| Initial assigment | not necessary | necessary |
| Seeking closest centroid | not necessary | necessary |
| Update strategy | incremental | egg-chicken loop |

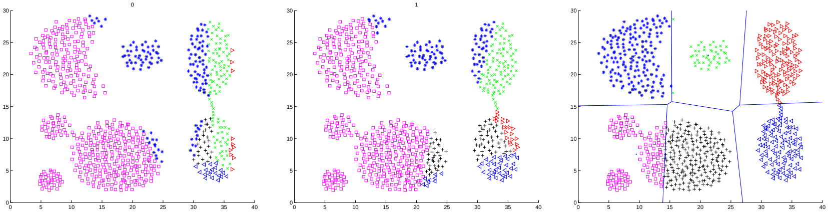
1. It is not necessary that assigns samples to closest initial centroid
2. It is not necessary to assign sample to its cloest centroid in the loop
3. Clusters are updated as soon as we find the moving is approperiate

# $k$-means$^{\#}$: a demo (1)




(a) initial (b) iter=1 (c) iter=10

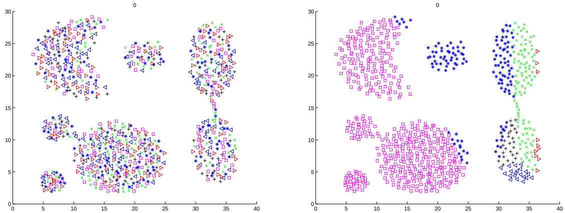
Figure 1. $k$-means$^{\#}$ iteration




(a) initial (b) iter=2 (c) iter=30

Figure 2. $k$-means iteration

# $k$-means$^{\#}$: a demo (2)




(a) $k$-means$^{\#}$ (b) $k$-means

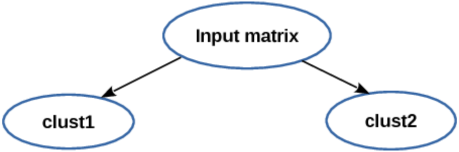
Figure: Comparison of initial assigment of two algorithms

# $k$-means$^{\#}$ in Animation

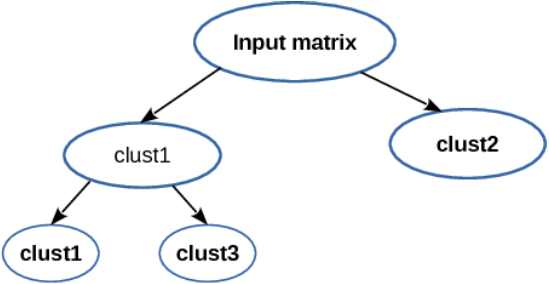
# Hierarchical $k$-means$^{\#}$ (1)

- $k$-means$^{\#}$ is faster than traditional $k$-means
- However, they are on the same complexity level: $O(n \cdot d \cdot k \cdot t)$
- We can perform $k$-means in a hierarchical manner
  1. Cluster given matrix into **2** clusters
  2. Pick an intermediate cluster
  3. Cluster the cluster into **2**
  4. Repeat **Step 2-3** until **k** is reached

# Hierarchical $k$-means$^{\#}$ (2)



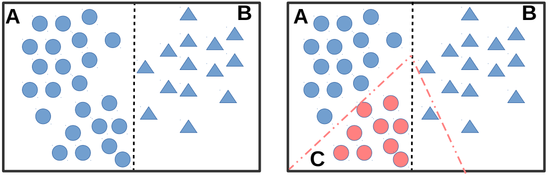

(a) the 1st partition




(b) the 2nd partition

# Hierarchical $k$-means# (3)

- The complexity of hierarchical clustering is $O(n \cdot d \cdot log(k) \cdot \overline{t})$
- Notice that $log(k)$ is much smaller than $k$
- That means $n \cdot d$ is multiplied by a small factor
- However, hierarchical $k$-means# faces underfitting problem




(a) the 1st round bisecting (b) the 2nd round bisecting

- Later, we will see the efficiency of hierarchical $k$-means# and its quality

# Experiment: clustering quality (1)

- We check how the original target is reached

$$Min. \sum_{q(x_i)=r} \| C_r - x_i \|^2 \tag{4}$$

- The final function score (distortion) is averaged

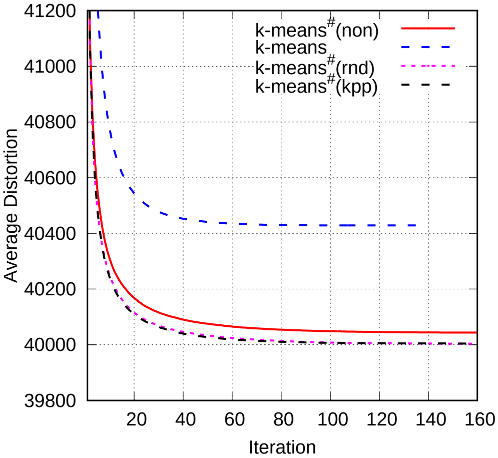
$$\Downarrow$$

$$\bar{E} = \frac{\sum_{q(x_i)=r} \| C_r - x_i \|^2}{n} \tag{5}$$

- The lower the better

# Experiment: clustering quality (2)

- Check the effect of initial assignment

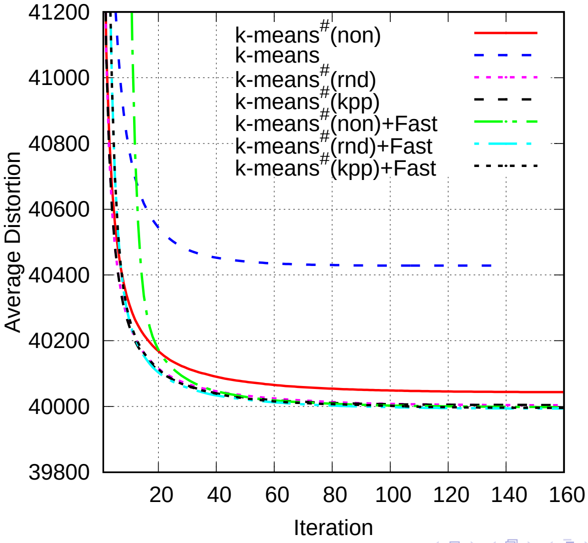



- The way of assigning samples to initial centroids
  - non: no initial assigment
  - rnd: same as *k*-means
  - kpp: same as *k*-means++

- Initial assignment impacts little to the clustering quality

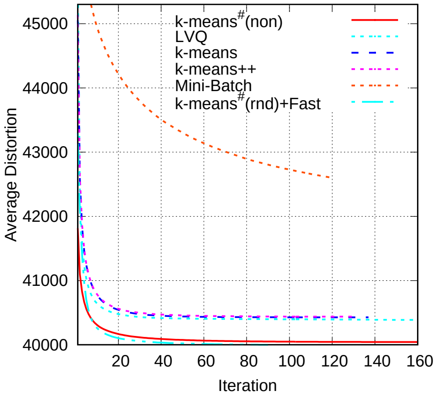
# Experiment: clustering quality (3)

- Check whether it is necessary to seek the best moving

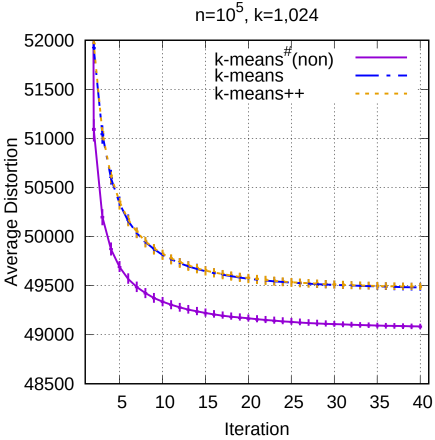


# Experiment: clustering quality (4)

- In comparison to *k*-means and its variants




(c) Comparison to *k*-means variants




(d) Significance test

- *k*-means# outperforms *k*-means and its variants considerably
- Verified by 128 runs plotted in candle chart

# Experiment: document clustering (1)

- **15** document datasets are adopted[2]
- Document is represented by vector under TF/IDF model
- Entropy is adopted for evaluation

$$Entropy = \sum_{r=1}^{k} \frac{n_r}{n} \frac{1}{\log c} * \sum_{i=1}^{c} \frac{n_r^i}{n_r} * \log \frac{n_r^i}{n_r}, \tag{6}$$

- In the range of [0,1], the lower the better
- The performance is averaged over 15 datasets

[2] http://glaros.dtc.umn.edu/gkhome/fetch/sw/cluto/

# Experiment: document clustering (2)

Table: Clustering performance (average entropy) on 15 datasets

| | k = 5 | k = 10 | k = 15 | k = 20 |
|---|---|---|---|---|
| *k*-means | 0.539 | 0.443 | 0.402 | 0.387 |
| *k*-means++ | 0.550 | 0.441 | 0.403 | 0.389 |
| Mini-Batch | 0.585 | 0.488 | 0.469 | 0.475 |
| KM$^{\#}$(non) | 0.552 | 0.442 | 0.388 | 0.368 |
| KM$^{\#}$(rnd)+Fast | **0.506** | **0.419** | **0.380** | **0.353** |
| BsKM | 0.532 | 0.438 | 0.410 | 0.373 |
| BsKM++ | 0.507 | 0.422 | 0.400 | 0.379 |
| BsKM$^{\#}$(non) | 0.514 | **0.388** | **0.353** | **0.329** |
| RBK | **0.486** | 0.402 | 0.366 | 0.339 |

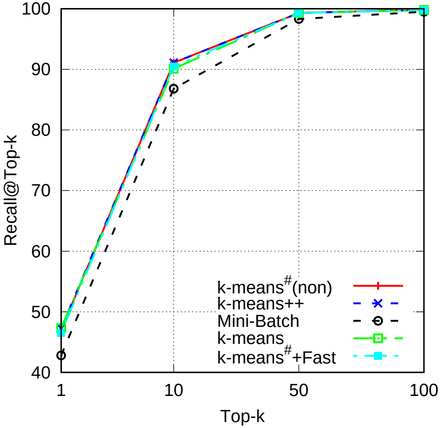
- Different numbers of clusters have been tested

# Experiment: product quantization (1)

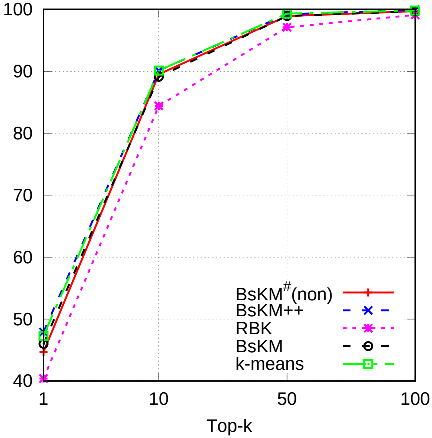
- Different clustering methods are adopted to produce the PQ vocabulary
- SIFT1M is adopted[3]
- 128-dimensional SIFT is PQ into 8 segments, each is encoded by 256 words
- The success rate of top-k nearest neigbor search is evaluated

[3] http://corpus-texmex.irisa.fr/

# Experiment: product quantization (2)




(a) direct k-way, m=16



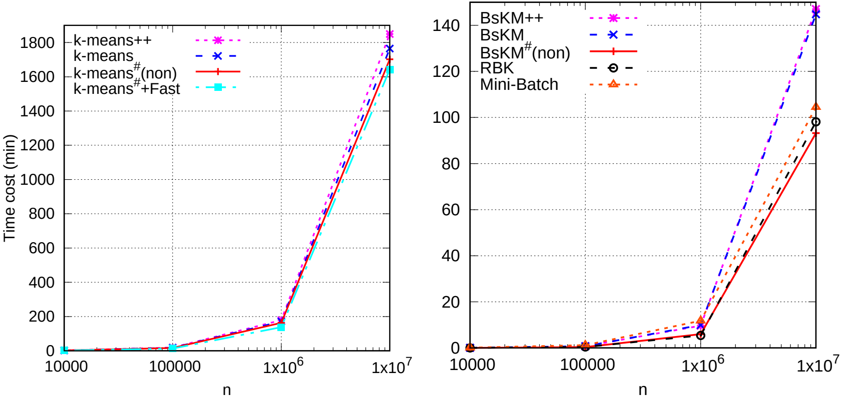

(b) bisecting, m=16

- PQ is tolerant to clustering quality
- However, Mini-batch and RBK (Repeated Bisecting *k*-means) are considerably poorer

# Experiment: image clustering (1)

- In order to test the scalability of $k$-means$^{\#}$
- 10M image dataset is adopted
- Image is represented as 512-dimensional VLAD vector
- We consider both clustering speed and quality (average distortion)

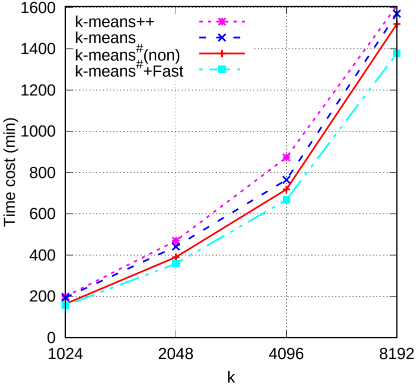
# Experiment: image clustering efficiency (1)



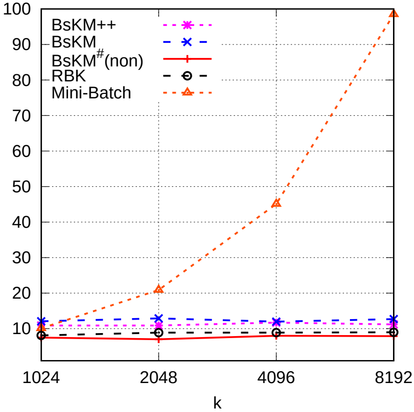

(a) k=1,024, direct k-way (b) k=1,024, bisecting

- *k*-means# is the fastest in two cases
- Bisecting is around 20 times faster than direct k-way

# Experiment: image clustering efficiency (2)



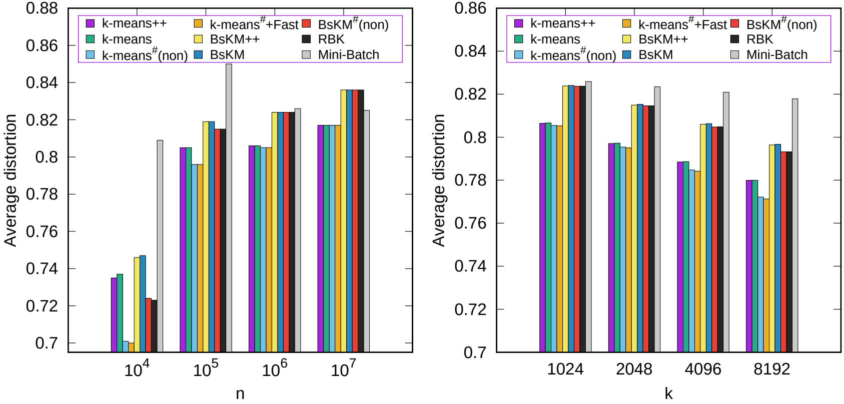

(a) $n = 10^6$, direct k-ways




(b) $n = 10^6$, bisecting

- When we increase the number of clusters
- $k$-means$^{\#}$ maintains its efficiency

# Experiment: image clustering quality




(a) k=1024, vary n (b) $n=10^6$, vary k

- $k$-means# achieves the best performance in direct k-way and bisecting cases

## Summary

- $k$-means$^{\#}$ is simpler
  - No chicken-egg loop
  - Initial assignment is not necessary
  - Moving to closest centroid is not necessary
- $k$-means$^{\#}$ always leads to lower clustering distortion
- $k$-means$^{\#}$ is the most efficient
- Story behind this work
- Source codes are available[4]
  - Motivated by the image linking problem
  - My student, **Chenghao Deng** suggested to remove the initial assignment

[4] https://github.com/wlzhao22/xkmeans

1. Empirical and Theoretical Comparisons of Selected Criterion Functions for Document Clustering, Ying Zhao and George Karypis, Machine Learning, 2004
2. *k*-means++: the advantages of careful seeding, D. Arthur and S. Vassilvitskii, 2007
3. Top 10 algorithms in data mining, X. Wu, V. Kumar and et al. Knowledge and Information Systems, 2008, 14(1): 1-37
4. The Nature of Statistical Learning Theory, Vladimir N. Vapnik , Springer-Verlag, 1995.
5. *k*-means: a revisit, W.-L. Zhao, C.-H. Deng, C.-W. Ngo, Neurocomputing, 2018
6. Lecture notes on Machine Learning, Andrew Ng., http://cs229.stanford.edu/

Q & A

Thanks for your attention!