Clustering Methods without Given Number of Clusters

Peng Xu, Fei Liu

1 Introduction

As we know, kmeans method is a very effective algorithm of clustering. Its most powerful feature is the scalability and simplicity. However, the most disadvantage is that we must know the number of clusters in the first place, which is usually a difficult problem in practice. In this paper, we propose a new approach–peak-searching clustering– to realize clustering without given the number of clusters.

Our method is based on the similarity graph[1] of the data points. Through the relationships between points, we capture those points which are near the cluster centers. By finding the peak area of the stationary distribution of the random walk on the corresponding similarity graph, we figure out a way to capture the points near the cluster center areas, which we call "peak points" in this paper. The advantage of our method is that we don't need the number of clusters as an input – our algorithm estimate the number of clusters in the dataset, which we believe can be a good indictor of the true value.

2 Peak-searching clustering

Denote $X := \{x^{(i)}\}_{i=1,\dots,m}$ as the sample data set. m is the sample size, n is the dimension, $x^{(i)} \in \mathbb{R}^n$. Since we do not know how many clusters there are in X, We try to have a good estimation of the number of clusters or the possible cluster centers. Assume the data set is dividable, then the points in one cluster tend to be near to each other. Our simple strategy is to capture those points which are near the cluster centers.

2.1 degree and stationary distribution

Consider the similarity graph $G = (V, E)$, with $V = \{x^{(1)}, x^{(2)}, \dots, x^{(m)}\}$. And $W \in \mathbb{R}^{m\times m}$ is the corresponding weighted adjacency matrix. The generalized degree of a point $x^{(i)}$ is defined as $d_i = \sum_{j=1}^{n} W_{i,j}$. The degree matrix is defined as $D = \text{diag}(d) = \text{diag}(d_1, d_2, \dots, d_m)$. Here, d_i is determined by i, or by $x^{(i)}$ – there's no essential difference. In the following passage we may use $d = d(x^{(i)})$ to denote the mapping $x^{(i)} \to d(x^{(i)})$ to simplify the illustration. Similarly, $W_{i,j}$ has no difference with $W(x^{(i)}, x^{(j)})$, and i can be used to denote the point $x^{(i)}$ – it won't cause any ambiguity.

Now, consider the random walk on graph G with weighted adjacency matrix W. Then, the random walk has a stationary distribution π over all the points. It's reasonable to claim that, the closer to the cluster center a point i is, the higher π_i is. That is to say, π_i is a "local maximum" of the stationary distribution if i is the "closest" point to the center. From the theories in stochastic process, we know that, π is the solution of the linear equation:

$$\pi P = \pi, \text{ subjected to } \sum_{i=1}^{m} \pi_i = 1$$

where $P = D^{-1}W$ is the corresponding transition matrix.
Since G is symmetric, we can directly solve π as $\pi \propto d$. So we can use d to indicate the relative value of π.

2.2 peak-searching process

For a well-defined clustering problem, there should be several peaks in the set $\{(x^{(i)}, d(x^{(i)})) : i = 1, 2, \dots, m\}$, that is, if we consider the mapping d from $\{x^{(1)}, x^{(2)}, \dots, x^{(m)}\}$ to $\mathbb{R}$, then the mapping should have several "local maxima" areas, which are the center areas of the clusters. Given the degree of all the points, we want to capture one point per such area, to locate one cluster center. We call these points the "peak points" of clusters. To find such points, first, we get the point with the highest degree as the first peak point. Note that, peak points should not be close to each other because each of them is near the center of different clusters, therefore, if we have an appropriate neighborhood of the first peak point, then the point of highest degree outside the neighborhood will be the second peak point. Theoretically, we can capture all the rest peak points by cutting off "appropriate" neighborhoods of the existing peak points. But it is difficult to determine the size of neighborhoods. Here we introduce the concept of persistency of points. As we increase the size of neighborhood, the highest degree outside the neighborhood will decrease. Specifically, we consider a k-nearest neighborhood of the first peak point. And as we increase k from 1 to m, more and more points are included in the neighborhood. The highest degree outside the growing neighborhood is decreasing. But there exist some resistance against such drop tendency. We illustrate this by an 1D example, Figure 1.

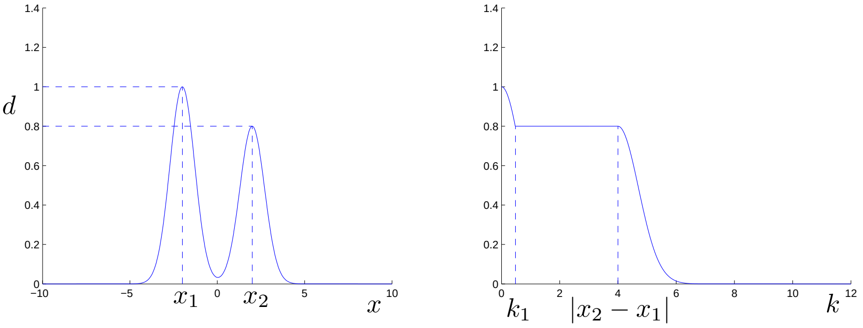


Figure 1: In the left figure, there are two peaks and we need to capture the peak points x_1,x_2. We get x_1 as the first peak point, because it has the maximum value of d. Then we continuously remove the neighborhood of x_1, which means we remove $(x - k, x + k)$ with k growing from 0 to ∞. As k grows, the maximum value outside the interval $(x - k, x + k)$ drops, as shown in the right figure. But it drops to $d(x_2)$ at $k = k_1$ and keeps the level until x_2 is included in the neighborhood of x_1 and then the value goes down again. In this example k is continuous, but the basic idea is the same for discrete situations.

We call such resistance against the drop tendency as the **persistency** of a point. In the example above, as we cut off the neighborhood of x_1 by larger and larger size, x_2 shows up to resist the drop tendency. The persistency of x_2 is defined as $|x_2 - x_1| - k_1$, which means the "length of period" it holds the maximum value of d outside the neighborhood of x_1. Similarly we can define persistency to any other point. To simplify our illustration, we call the maximum value of d outside the k-nearest neighborhoods of some points as $s(k)$. s also depends on the "some points", but for simplicity, we just use $s(k)$ if it doesn't cause ambiguity.

Peak points tends to have high persistency than non-peak points. In the example above, x_2 has the highest persistency, so we pick it out as the second peak point.

After we get N peak points. We search the $(N+1)$-th peak point (if there is any) by the following rules: We cut off the k-nearest neighborhoods of the N current peak points simultaneously and observe the point with the highest d among the rest of the points, then we pick out the most persistent point during the growth of k from 1 to m. Then, the point we pick out is the potential $(N + 1)$-th peak point.

Now we consider the termination of the searching process. One way is to set a lower bound for the persistency, since short persistency does not reflect the stable structure. But the lower bound varies between different data sets and it is not easy to select. Here we use another method. We preprocess the data to find out those points that are likely to be near the centers of clusters. Recall that we have d as the indicator of the likelihood of a point to be near the center. If d_i is higher than most of the d_j where j is near i, then d_i is more likely to be a center. To compare d_i with d_j where j is near i, we simply compare it with the average weighted value of d_j, where the weight is $W_{i,j}$, an indicator of how j is near i. We call such average weighted value as h_i, then we can compute h by:

$$h = dP^T = dWD^{-1}$$

Then, the termination condition is:
For a potential peak point $x^{(i)}$, if $d_i > h_i$, then i can be considered as a real peak point, and the searching process can proceed; if it's not, then i is not considered as a real peak point, and the process terminates.

The idea of finding h comes from the theory related to heat equation: h is indeed a smoothness of d by convoluting d with some probability distributions. It "drags down" the peaks and "raises up" the valleys. The behavior of heat is the same: as the time goes on, heat flows from position with high temperature to position with low temperature. And the solution to the heat equation is a convolution of the initial data with the Poisson heat kernel. If we want to know where the peaks are, we simply find where the values are dragged down. And the “dragged down” area should be the peak area.

Algorithm 1 Peak-searching clustering

```
Input:
  • {$x^{(i)} : i = 1, 2, \ldots, m$}, input data
  • $W$, weighted adjacency matrix
Output:
  • peak points (cluster centers)
  • cluster labels of all points
Pseudo code:
  1. Compute $d_i = \sum_{j=1}^{m} W_{i,j}$
  2. Compute $h_i = \sum_{j=1}^{m}(\sum_{j=1}^{m} W_{i,j} d_j)/d_i$
  3. Add $x^{(i)}$ which has the highest $d_i$ into the set of peak points
  4. Find all peak points, repeat until stop:
       • Set the persistency of all points to 0
       • for $k = 1, 2, \ldots, m$
           – Find $x^{(i)}$ which has the highest $d_i$ among those points that are not in the $k$-nearest neighborhoods of all the current peak points
           – Let the persistency of $x^{(i)}$ increase by 1
         end
       • Find $x^{(c)}$ which has the highest persistency
       • if $d_c > h_c$
           – add $x^{(c)}$ into the set of peak points
         else
           – stop finding peak points
         end
     end
  5. Set the peak points as the cluster centers. Let each of the other points shares the same cluster label as its nearest peak point.
```

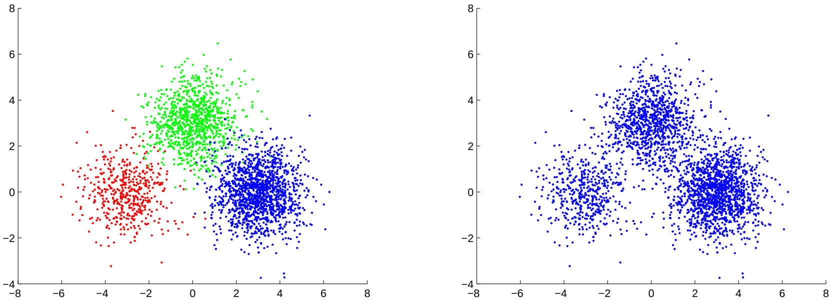
2.3 clustering

After getting all the peak points, we have the number of clusters as the number of peak points. Noting that the peak points are quite close to the cluster centers, we can directly regard the peak points as the cluster centers and all the other points are assigned to different clusters based on the distance between the points to the peak points. The point shares the same cluster label with the nearest peak point.

3 Experiments

In this section, we will use both simulated data and real data to demonstrate the utility of our approach. In the real data experiments, we compare our methods with dpmeans method[2] and kmeans method. Throughout the experiments, we use normalized mutual information(NMI) between the ground truth classes and algorithms outputs for evaluation. When using kmeans, we use our estimation of the number of clusters as the input. As to dpmeans, we apply max-min random selection method to estimate λ based on our estimation of the number of clusters.
Construction of Graph: We can either use mutual k-nearest neighbor graph or Gaussian weighted graph in our experiments. In this paper, we use Gaussian weight, we choose the parameter σ^2 in Gaussian similarity function to be the mean variance of the original data. We also have done experiments with mutual k-nearest neighbor graph and generally we can achieve good performance when setting $k \approx 15\%m$, the result is not shown in this paper.

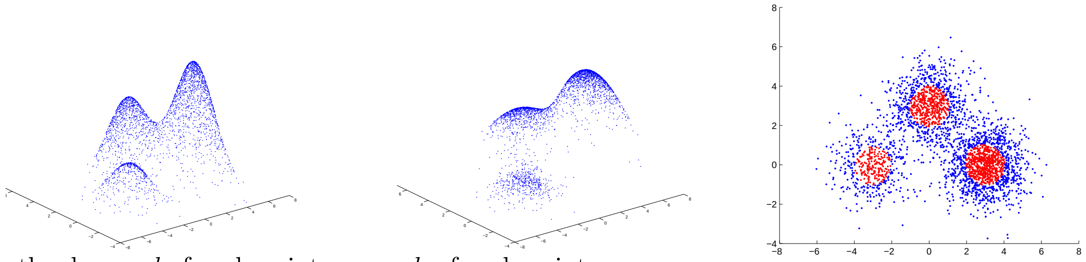
First, we use 3 simulated sets data of Gaussian distribution on the 2D plane to show how our algorithm works (Figure 2,3,4,5). We also apply our method to 5 UCI data sets. For each set of real data, we first apply PCA to the original data(keeping over 98% principle components), and then implement our clustering methods. The results are shown in Tabel 1.



data with label

data without label

Figure 2: The left figure is the original sample points generated by 3 Gaussian distributions with covariance matrix equal to I, and mean equal to $(-3, 0)$ for the red-colored set, $(0, 3)$ for the green-colored set, and $(3, 0)$ for the blue-colored set. The number of sample points in each colored set is: 500 red, 1000 green, 1500 blue.

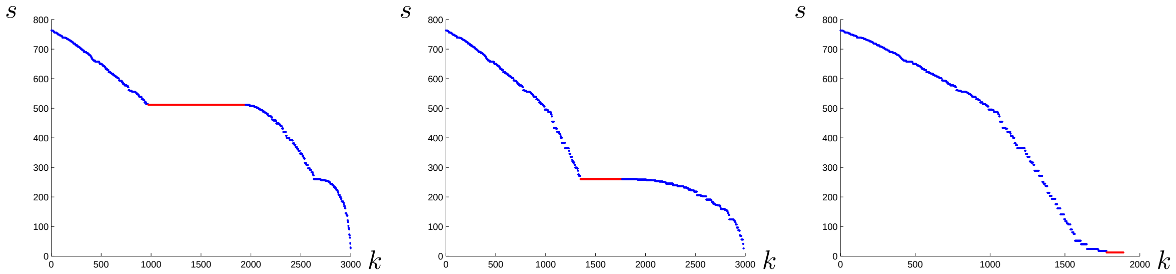


the degree d of each point

h of each point

the comparison of d and h

Figure 3: In the left figure, d represents the stationary distribution of the random walk on the graph G, the value of d successfully reflects how much a point is near a cluster center – the peak areas in d are exactly the center areas of clusters. The middle figure shows h, a smoothness of d. In the right figure, we mark the points with $d > h$ as red, and $d < h$ as blue, we see that such standard is a good indicator of whether or not a point is in the center area of a cluster.

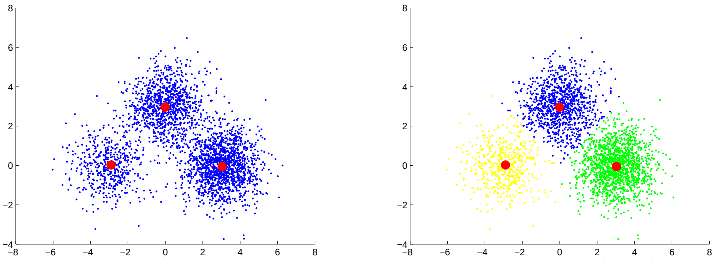


persistency, with 1 peak point

persistency, with 2 peak points

persistency, with 3 peak points

Figure 4: These three figures shows the value of $s(k)$ of the peak points. $s(k)$ drops as k grows, but as we can see, there are points that "resist" such drops. In the figure, we marked out with red line the largest persistency period. In the first figure, we use 1 peak point (the point with highest value of d) and keep cutting off its k-th nearest neighborhood, and capture the second peak point by its persistent behavior. In the middle figure, we use peak point 1 and 2 simultaneously to cut off their k-th nearest neighborhoods and capture the third peak point. In the right figure, As we've already found 3 peak points, there's no significant persistency anymore, what we get is a fake peak point. By the corresponding value of h and d, it's not in the center area and the searching process terminates.



peak points

assigned labels

Figure 5: As the searching for peak point terminates, we can assign the cluster labels to all the points. The left figure shows the peak points found, which is marked as red. The right figure shows the assigned labels.

data set	PSC	dpmeans	kmeans
Iris(3)	0.7208(4)	0.7744(4)	0.7219(4)
Wine(3)	**0.4345(2)**	0.4233(4)	0.4259(2)
Seeds(3)	**0.6987(3)**	0.4857(2)	0.6949(3)
Soybeans(4)	**0.7375(6)**	0.6930(5)	0.7103(6)
Pima(2)	**0.0517(4)**	0.0219(5)	0.0264(4)

Table 1: UCI data sets:NMI(number of clusters). In the first column are the names of data sets and the numbers in the parentheses are the numbers of true classes. In the other columns, the numbers in the parentheses are the numbers of clusters algorithms output. In the case of kmeans, it is same with that of peak-searching clustering(PSC).

4 Discussion

In the previous sections, we have exhibited a brand new clustering method – peak-searching clustering. It has good performances in many cases. There are a few points we should mention here.

First, our method is based on the similarity graph, and essentially based on the Euclid distances of data points. In this case, our method can only deal with linearly dividable problems, the same with kmeans approach. For the structures like “a ring within a ring”, we cannot seperate the two rings. Second, PSC is a very intuitive and straightforward method. We start from the connection between points and try to distinguish points lying in the centers with those lying on the margins. We give reasonable clusters based on those center points. However, as to clustering problem itself, the number of clusters is probably undeterminable in practice, since there is no absolute standard of clustering. A good clustering method can only provide reasonable solutions instead of the right solutions, which is exactly what our approach has done. The last point is the parameter selection problems. As in PSC, the construction of similarity graph turns out to be a very important step. Either using k-NN graph or Guassian weighted graph, we have parameters k or σ to be determined. In some extreme cases, the clustering result is quite sensitive to the parameter selection. In our experiments, we choose σ^2 as the mean of variances of the data points in all attributes. Probably this is not the best choice. How to select σ might be a good problem to work on.

5 Conclusions

In this paper, we start from a simple intuition, and provide a new clustering method without knowing the number of the clusters – peak-searching clustering. We explain our ideas from the perspective of random walk. We also introduce the persistency concept in the peak-searching process. And in the experiments, PSC does a good job in clustering problems.

References

[1] U. von Luxburg, A tutorial on spectral clustering, Tech. Rep. 149, Max Planck Institute for Biological Cybernetics, August 2006.

[2] Kulis, B. and Jordan,M.I. Revisiting k-means: New Algorithms via Bayesian Nonparametrics. In *ICML*,2012.