K-SEARCH: SEARCHING FOR CLUSTERS

Rhonda Phillips and Bijaya Zenchenko

MIT Lincoln Laboratory
Lexington, Massachusetts 02420 USA
{rhonda.phillips,b.zenchenko}@ll.mit.edu

ABSTRACT

This paper introduces the K-search algorithm, a method for locating an unknown number of well-separated multidimensional clusters from sampled data in the presence of outliers. K-search finds tightly packed point clouds, a characteristic of Gaussian data close to a mean value, to identify potential Gaussian means. Using this search strategy, the approximate locations of cluster means are found, automatically providing an estimate for the number of clusters, K. In experimental results, K-search can effectively identify the true number of well-separated Gaussian clusters and their locations in the presence of random background clutter (outliers). We use K-search to identify modal driving behaviors in a real vehicle track dataset in the presence of noisy tracks, and we compare results to other model based clustering methods that automatically determine K.

1. INTRODUCTION

Clustering, the unsupervised process of learning structure from data, has traditionally been a problem of associating N data points to K clusters. Unfortunately, finding an optimal clustering for a given dataset is often computationally intractable. Many algorithms have been introduced to address specific clustering issues. Two of these issues include determining the correct number of clusters and obtaining a good clustering in the presence of noisy data.

Center based algorithms such as K-means and Gaussian mixture models (GMMs) are efficient and scale well for large datasets, but require initialization and therefore find local optima. Finding the correct number of clusters is difficult because objective functions that minimize the error between clusters and the points they represent can always be reduced by adding more clusters, and at the extreme, errors will be zero when each point is its own cluster. The presence of outliers further complicates these issues as model assumptions are violated, resulting in finding incorrect clusters.

In this paper, we address both issues by finding the correct number of Gaussian or Gaussian-like clusters in the presence of non-Gaussian outliers. We first identify local regions of large point density in order to (one) learn the number of clusters and (two) learn the approximate location of those clusters. We then use a center based clustering method such as K-means to converge to the locally optimal clustering based on our initialization. Given the correct number of clusters and initial cluster centers close enough to the real cluster centers, the local optimum value found by K-means, etc., will be globally optimal. This approach addresses one of the biggest drawbacks of K-means and other center based approaches (initialization), while retaining some of the most desirable properties (scalability and efficiency). Other model based methods that determine K are sensitive to outliers that violate model assumptions. The major contribution of this paper is the introduction of K-search, an algorithm that searches for Gaussian data in a multidimensional feature space that includes outliers. Additionally, we provide evidence that under certain conditions, our search algorithm will locate the true Gaussian clusters. We demonstrate that K-search can locate clusters in the presence of outliers (data that does not belong to any clusters). This is a significant contribution to clustering as current cluster initialization methods (or cluster finding methods) use the entire dataset to learn clusters, whereas our method is robust to the presence of background clutter.

This work was sponsored by the Department of Defense under Air Force Contract FA8721-05-C-0002. Opinions, interpretations, conclusions, and recommendations are those of the authors and are not necessarily endorsed by the Department of Defense or the United States Air Force. Approved for public release; distribution is unlimited.

2. RELATED WORK

Robust clustering is a term that describes clustering methods that are robust to the presence of noisy data or outliers. All of these methods require selecting model parameters such as the number of initial clusters. Khan and Ahmad proposed an algorithm for cluster center initialization assuming K is known [1]. Their approach is based on the experimental observation that individual features can provide some insight into the initial cluster centers. The first step of the algorithm is the computation of cluster centers for individual features using the K-means algorithm. In order to combine clusters found using individual attributes, they use centers that are far apart while removing outliers.

Hamerly and Elkan proposed the G-means algorithm that attempts to learn K and intelligently initialize clusters [2]. G-means accepts individual clusters that follow Gaussian distribution and splits clusters that do not using principal components analysis (PCA). G-means runs K-means with increasing K in a hierarchical fashion until all clusters are sufficiently Gaussian. This method does not assume spherical clusters and works best if the true clusters are well-separated.

Feng and Hamerly proposed an updated version of G-means, PG-means, which is able to learn the number of clusters in a classical Gaussian mixture model (GMMs), instead of K-means [3]. To that end, they apply a statistical test to the entire model at once, not just on a per-cluster basis. Their method works better than G-means for overlapping data and other scenarios where data are not Gaussian. However, PG-means, like G-means, learns the positions (and number) of clusters using all of the data contained in the dataset. Feng and Hamerly demonstrated that PG-means produces clusters comparable to Bayesian K-means (using Maximization-Expectation to learn a GMM) [4] in significantly less time [3].

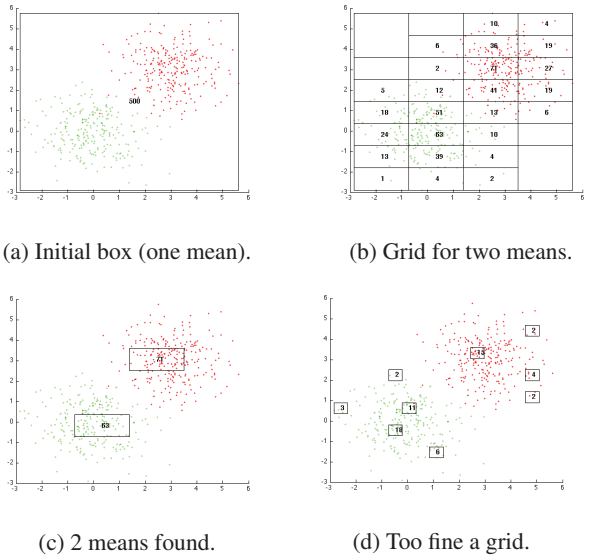
Both G-means and PG-means use a principled, model based approach to locate clusters, but the model assumptions are violated by the presence of outlying background clutter. Our clustering method uses similar model assumptions, but the way we locate potential clusters is robust to outliers. Instead of recursively splitting a cluster that does not fit a particular model, we search for sets of points that are closely clustered around a center with less point density as the distance from the center increases, a property of Gaussian data (and other distributions). Like PG-means, a mixture of Gaussians is assumed to properly model the data and test for model fit. Performance of K-search will be compared with that of G-means and PG-means using pristine synthetic data, synthetic data in the presence of uniform outliers, and a real noisy dataset of vehicle tracks.

3. ALGORITHM

K-search locates data such as Gaussian data that is characterized by density near a mean value. The main idea behind the algorithm is that by discretizing the search space, it is possible to calculate the density of points within each grid cell; cells that contain mean values will tend to have larger point densities than neighboring cells without mean values. As the size of each grid cell decreases, there is more likely to be grid cells separating mean values, and therefore grid cells that have higher point density than neighboring cells will indicate the presence of a mean (or mode).

K-search begins with exactly one grid cell encompassing the entire dataset. The multidimensional hyperbox is defined by a minimum coordinate and a maximum coordinate that correspond to minimum and maximum values in the dataset. By default, that one box is denser than neighbors (it has no neighbors), and the resulting model describing the data is represented by the one box. The multidimensional grid cell is then split into two halves along the largest dimension of the hyperbox, and both hyperboxes are tested for density relative to neighboring boxes (each other). The algorithm continues to divide cells and test for density until cells have no more than 1 point (a more practical lower bound could be set in practice). Note this hierarchical structure of grid cells is very similar to that of kd-trees, although unlike kd-trees the grid cells are uniform in size at a particular grid level.

In order to make this method practical for real Gaussian or Gaussian-like data, there needs to be a selection criteria for the appropriate grid size. Figure 1 shows the cell dividing method applied to 500 points randomly drawn from two two-dimensional Gaussian distributions. Figure 1a shows the initialization with one hyperbox, and Figure 1b shows the grid when two means are found (shown in Figure 1c). Figure 1d demonstrates the potential means that are found when too fine a grid is used. Notice that once the grid level is too fine, multiple dense boxes are found for each true cluster, an artifact resulting from having a finite number of samples. At the extreme, when the grid is small enough that each cell only contains one point, several of those singleton cells would be dense compared with neighboring cells that do not contain any points. To address this issue, we propose a hypothesis test to determine if the points observed within the current set of dense boxes were more likely to have been caused by the previously accepted model (mixture of Gaussians), or are more likely to have been caused by the new model described by the new set of dense boxes. If small artificial dense sets are located in a true cluster, the model including the true cluster will have higher probability (given all of the observed data) than the fragmented data.



(a) Initial box (one mean). (b) Grid for two means.

(c) 2 means found. (d) Too fine a grid.

Fig. 1. K-search algorithm looking for 2 Gaussian clusters.

In order to test whether the data realized at the latest grid level are more likely to be drawn from $\Theta^{(k-1)}$ or $\Theta^{(k)}$ we form two hypotheses:

$$H^{(k-1)} : D \sim \Theta^{(k-1)}, \quad H^{(k)} : D \sim \Theta^{(k)}.$$

$\Theta^{(k)}$ is estimated by the data $x_i \in \Re^B$ by first assigning each x_i to the nearest dense box using the Mahalanobis distance, and then estimating necessary parameters for each Gaussian from the assigned data. All points x_i in the dataset are assigned to one of the $G^{(k)}$ potential Gaussian clusters (one for each dense box) at step k by minimizing the Mahalanobis distance,

$$d_i = \underset{j}{\operatorname{argmin}} \sqrt{(x_i - c_j)\Sigma_j^{-1}(x_i - c_j)^T},$$

forming the partition $d \in \Re^B$ where $c_j \in \Re^B$ is the centroid of box j and Σ_j is a diagonal matrix with half the total width of box j in each dimension on the corresponding diagonal (allowing for the potential of different sized hyperboxes when inserting newly found clusters into the previously found model). J_g is the index set such that $i \in J_g$ implies $d_i = g$.

The model used for the hypothesis test,

$$\Theta^{(k)} \equiv \{g = 1 \ldots G, \hat{q}_g^{(k)}, \hat{\mu}_g^{(k)}, \hat{\sigma}_g^{(k)^2}\},$$

uses estimated parameters for the mixing proportions $\hat{q}_g^{(k)} \in \Re$, the Gaussian mean vector $\hat{\mu}_g^{(k)} \in \Re^B$, and the Gaussian standard deviation $\hat{\sigma}_g^{(k)} \in \Re^B$. The parameters of the gth distribution are estimated from the set of points assigned to the gth centroid c_g,

$$\hat{q}_g^{(k)} = \frac{n_g^{(k)}}{N}, \quad \hat{\mu}_g^{(k)} = \frac{1}{n_g^{(k)}} \sum_{i \in J_g^{(k)}} x_i, \text{ and}$$

$$\hat{\sigma}_g^{(k)} = \sqrt{\frac{1}{n_g^{(k)} - 1} \sum_{i \in J_g^{(k)}} (x_i - \hat{\mu}_g^{(k)})^2}.$$

Once each distribution is estimated from the assigned data, the probability of each dense hyperbox (recall a hyperbox is the multidimensional space described by a minimum coordinate $min_g^{(k)} \in \Re^B$ and a maximum coordinate $max_g^{(k)} \in \Re^B$) can be calculated using the multivariate normal cumulative probability function, $\Phi(t|\mu, \sigma^2)$,

$$\pi_j^{(k)} = \sum_{g=1}^{G} \hat{q}_g^{(k)} [\Phi(max_j^{(k)} | \hat{\mu}_g^{(k)}, \hat{\sigma}_g^{(k)^2})$$

$$-\Phi(min_j^{(k)} | \hat{\mu}_g^{(k)}, \hat{\sigma}_g^{(k)^2})].$$

The sum of individual dense box probabilities is the probability of drawing a sample in any of the dense boxes, $\pi^{(k)} = \sum_{g=1}^{G} \pi_g^{(k)}$, Using this probability, we can calculate the probability of finding the number of samples $m^{(k)}$ in all of the dense boxes given that we have a total of N samples and probability $\pi^{(k)}$ using the binomial probability distribution function $\Psi(m|\pi, N)$, $P(D|H^{(k)}) = \Psi(m^{(k)}|\pi^{(k)}, N)$, where $m^{(k)} = \sum_{g=1}^{G} m_g^{(k)}$ and $m_g^{(k)}$ is the number of samples in the gth box at the kth step.

In addition to those probabilities, there are terms in the likelihood ratio test for the prior probabilities of a hypothesis, $P(H^{(k)})$. In this case, we assume a uniform distribution for the prior probability of a cluster being in any given location in our bounded multidimensional space. Therefore the prior probability is equal to the sum of the volume of the dense boxes over the volume of the entire bounded space (the first box):

$$P(H^{(k)}) = \frac{\sum_{g=1}^{G^{(k)}} \prod_{i=1}^{B} \left(max_{g,i}^{(k)} - min_{g,i}^{(k)} \right)}{\prod_{i=1}^{B} \left(max_{1,i}^{(1)} - min_{1,i}^{(1)} \right)},$$

where $max_{g,i}^{(k)}$ is the ith maximum coordinate of the gth box for the kth step (and likewise for $min_{g,i}^{(k)}$).

The likelihood ratio test is

$$\textbf{if } \log \frac{P(D|H^{(k)})}{P(D|H^{(k-1)})} > \log \frac{P(H^{(k-1)})}{P(H^{(k)})} \textbf{ reject } H^{(k-1)}.$$

In the example shown in Figure 1 where two dense hyperboxes are located, $P(D|H^{(k-1)}) = 1E-14$, whereas $P(D|H^{(k)}) = .04$, leading the model containing the two means to be accepted. When the grid size is too small (Figure 1), $P(D|H^{(k-1)}) = 1E-6$ and $P(D|H^{(k)}) = 1.6E-5$. The likelihood of the data given the new model is not sufficiently larger than the likelihood of the data given the old model when prior information is taken into account, resulting in rejecting the model containing the 9 small clusters. The algorithm K-search follows.

Algorithm K-search.

1. Initialize grid and model.
2. Divide all cells with more than 1 point, test cells for density.
3. If (new means are found)
4. Construct new model, test new model against previous model, replace if more significant.
5. Repeat steps 2-4 until no dense cells are found.
6. Use model to initialize center based clustering algorithm such as K-means.

4. EXPERIMENTAL RESULTS

The K-search, G-means, and PG-means algorithms were run on randomly generated data to test the ability of each algorithm to locate the correct number of clusters. An α value of .0001 was used for G-means and an α value of .001 was used for PG-means. Additionally, each generated partition was compared to the true partition to evaluate the "correctness" of the clustering. This is challenging as the number of clusters can vary (when the true number of clusters is not recovered), and even when the number of clusters is identical, the individual clusters might not be. An information theoretic evaluation measuring how well the partitions match (agnostic of what the labels/numbers mean) is used. A pair of information theoretic metrics that have similar characteristics to probability of detection and probability of false alarm on the Receiver Operating Characteristic (ROC) curves were developed to evaluate trackers and classification results [5]. The information coverage score is analogous to probability of detection (truth information captured by learned partition) and the false information ratio score is analogous to the probability of false alarm (false information introduced by learned partition). A composite score defined as the information coverage minus the false information ratio is reported for experimental results, providing an indication of overall cluster correctness.

Gaussian clusters were generated by randomly selecting means using a uniform distribution and randomly selecting variances between one and two in each dimension (making the clusters elliptical in shape). Overlapping clusters (separated by less than 3σ) were allowed, but were discouraged by attempting to move one of the mean values if a move was possible. Table 1 contains results of these experiments.

Table 1. Experimental results on Gaussian data, IC Score is the information coverage score minus the false information ratio score as described in Section 4, d=dimension, IC=Information coverage score.

Algorithm	k	d	points	clusters detected	IC
K-search	5	2	1000	5±0	.99
G-means	5	2	1000	5.2±.41	.98
K-search	20	3	2000	19.75±1.55	.96
G-means	20	3	2000	20.55±1.36	.96
K-search	50	4	5000	50.7±2.15	.99
G-means	50	4	5000	51.3±1.13	.99

A second experiment was conducted with generated data where clutter was inserted into the otherwise Gaussian data using a uniform distribution. In addition to the specified number of Gaussian distributions, an additional uniform distribution (with proportion equal to one individual Gaussian distribution) was sampled, creating clutter not described by any of the Gaussian distributions. Experimental results are presented in Table 2.

Finally, we used our algorithm to identify modal driving behaviors on track posit output from a dataset of vehicle tracks. Each track posit has 2-dimensional velocity and 2-dimensional acceleration vector state estimates in addition to 2-dimensional position state estimates. The goal of clustering the track posits is to identify common driving behaviors based on velocity and acceleration. A 3-dimensional feature space was derived where the dimensions correspond to magnitude of velocity, magnitude of acceleration, and the angle between velocity and acceleration. An angle between ve-

Table 2. Experimental results on Gaussian data with uniform clutter, IC score is the information coverage score minus the false information ratio score as described in Section 4, d=dimension, IC=Information coverage score.

Algorithm	k	d	points	clusters detected	IC
K-search	5	2	1000	5.6±1.39	.91
G-means	5	2	1000	10.85±2.03	.76
PG-means	5	2	1000	6±0	.94
K-search	20	3	2000	19.95±.39	1.0
G-means	20	3	2000	47±7.9	.88
PG-means	20	3	2000	22.35±1.84	.98
K-search	50	4	5000	49.9±.97	1.0
G-means	50	4	5000	114.55±7.97	.90
PG-means	50	4	5000	44.2±2.91	.95

locity and acceleration ($d\theta$) between 0 and π indicates a turn to the left, an angle between π and 2π indicates a turn to the right, an angle of π is a deceleration, and an angle of 0 or 2π is an acceleration. These features enable discrimination between directional turns and different speeds, but does not consider heading. 10,000 track posits were clustered, and 2,000 were randomly sampled to determine the cluster centers using K-search. The final K-means clustering results are shown in Figure 2 in the 3-dimensional feature space. K-search identified 10 unique clusters. The 10 clusters all correspond to logical driving behaviors such as right turns and fast moving accelerators.

This clustering problem highlights the ability of K-search to find clusters in a noisy background as the track posit feature space is densely packed with points, but there are modal behaviors that can be identified by K-search. The modal behaviors are left turns, right turns, stops, slow rights, slow lefts, decelerations (no turns), fast decelerations, fast accelerations slightly to the right, fast accelerations slightly to the left, and large accelerations.

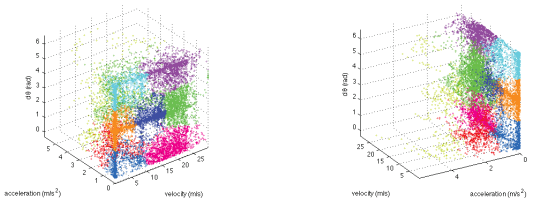


Fig. 2. Track posits clustered using K-search, K=10.

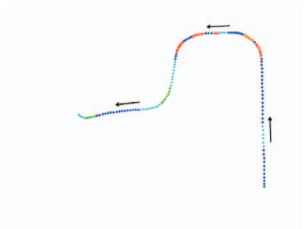


Fig. 3. Example track color coded by cluster value.

Experimental results in Table 1 illustrate that for low dimensional data tested in this paper, K-search on average performs as well as G-means or slightly better (results for PG-means are not reported as K-search and G-means both located the true clusters very well). In well-conditioned data without clutter, K-search will provide good initial clusters that result in a partition that captures the true partition. An advantage of K-search over G-means and PG-means is that no parameters (such as the significance threshold α) need to be set.

A more interesting feature of K-search is highlighted by results shown in Table 2, where uniform clutter is added to the Gaussian means. Because K-search seeks local dense point clouds in the feature space, the uniform clutter has little effect on the algorithm. The clutter, however, greatly affects the G-means algorithm because the location of new clusters depends on correlations in the underlying data (the cluster splitting step in the algorithm), and the addition of clutter affects the test used to determine if a particular cluster is Gaussian. The presence of clutter had a smaller but still noticeable effect on PG-means. The search mechanism and statistical test used in K-search are less rigid and do not break down in these situations when clutter is introduced. The IC metric indicates that K-search not only finds the correct number of clusters (or reasonably close), but those clusters are consistent with the true clusters.

5. CONCLUSIONS AND FUTURE WORK

This paper introduced the K-search algorithm that includes a novel approach to searching for Gaussian (like) means in a dataset by discretizing the search space and identifying potential means. Experimental results demonstrate that for low-dimensional Gaussian data, K-search performs as well as G-means, another algorithm designed to find separated Gaussian data, and in many cases performs slightly better. Furthermore, when clutter is introduced, K-search is virtually unaffected, whereas G-means and PG-means are worse at determining the correct number of clusters and capturing the true partition. This has further implications for real-world clustering problems where the goal is to find inherent structure in data in the presence of clutter. Future work includes modifications to the algorithm to make the approach more suitable for datasets with larger dimensions (the current algorithm does not scale well for those situations) and a local search mechanism that can modify the grid size to find different sized clusters.

6. REFERENCES

[1] S.S. Khan and A. Ahmad, "Cluster center initialization algorithm for k-means clustering," *Pattern Recognition Letters*, vol. 25, no. 11, pp. 1293–1302, 2005.

[2] G. Hamerly and C. Elkan, "Learning the k in k-means," in *Advances in Neural Information Processing Systems*, 2004, vol. 16, pp. 281–288.

[3] Y. Feng and G. Hamerly, "*pg*-means: Learning the number of clusters in data," in *Advances in Neural Information Processing Systems*, 2006, vol. 19, pp. 393–400.

[4] M. Welling and K. Kurihara, "Bayesian k-means as a "maximization-expectation" algorithm," *Neural Computation*, vol. 21, pp. 1145–1172, 2009.

[5] E.K. Kao, M.P. Daggett, and M.B. Hurley, "An information theoretic approach for tracker performance evaluation," in *Proc. IEEE International Conference on Computer Vision (ICCV)*, 2009.