K-Means and PSO based Clustering

Benjamin Lickiss Paresh Bhambhani

***Abstract*— Unsupervised learning can reveal the structure of datasets without being concerned with any labels, K-means clustering is one such method. Traditionally the initial clusters have been selected randomly, with the idea that the algorithm will generate better clusters. However, studies have shown there are methods to improve this initial clustering as well as the K-means process.This paper examines these results on different types of datasets to study if these results hold for all types of data. Another method that is used for unsupervised clustering is the algorithm based on Particle Swarm Optimization. For the second part this paper studies the classic PSO based algorithm and a Hybrid PSO algorithm which uses PSO to improve the results from K-means. The two PSO based algorithms are compared to the standard K-means clustering on two benchmark classification problems.**

I. Introduction

In unsupervised learning, the training methods do not use any forms of labels during the algorithms. This can shorten the time needed to train a classifier, and allows researchers to spot structures in the data. One of the methods in unsupervised learning is K-means method [4] [9], which classifies the data into k different clusters. Each cluster is assumed to be Gaussian and spherical, with each data point belonging to the cluster whose center it is closest to.

The traditional method for initializing the K-means method is to randomly assign cluster centers and let the algorithm distribute those random centers to appropriate locations. However, depending on the data structure this does not always create predictable clusters after training. A refined initialization method has been developed by Bradley and Fayyad[1] that refines the random initial clusters. The refined clusters are then used in the K-means algorithm to classify the data. The refined initial clusters are designed to generate more predictable clusterings.

Particle Swarm Optimization based clustering algorithm has been used for image and data vector clustering [3][10][13]. A hybrid PSO algorithm was proposed by DW van der Merwe and AP Engelhrecht in [13]. This paper will be comparing the hybrid PSO algorithm against the standard PSO and standard K-means (scikit package) algorithm[11], over Wine and Digits dataset.

II. Method Description

A. Refined K-means

The refined initialization method uses a set of J subsections of the data. Each of these subsections is meant to be a random pick of a small percentage of the original data. From each subsection a set of k cluster centers is found, with any empty clusters assigned to the point with the most distortion and then re-clustering the whole subsection. Once all subsections have non-empty k cluster centers, the set of $J * k$ points are clustered using a random initialization. The result of this clustering is used as the initial centers for the K-means clustering on the whole dataset.

In the Bradley and Fayyad paper which developed this process, the refined initialization was compared to the random initialization using 2 metrics. The first was average class purity, a measure based on the labels of the data. The second was the distortion, or sum of the L2 distance squared of the data, of the clusters where L2/Euclidean distance is given as:

$$\mathrm{d}(\mathbf{p}, \mathbf{q}) = \sqrt{\sum_{i=1}^{n} (q_i - p_i)^2} \tag{1}$$

In a similar manner, this paper uses the silhouette score[12] as a measure of quality. This score, from -1 to 1, compares the inter-cluster distance of data to the distance to the nearest cluster. A negative score represents mis-clustered data, with points assigned to a cluster that should be in another. A positive score represents defined clusters, with a higher score meaning more distinct clusters. A score of 0 represents overlapping clusters.

To truly investigate the difference between the random and refined initialization, and to compare PSO based algorithms with K-means, 3 different types of datasets were compared. From the UCI Machine Learning Repository[8] the Wine, Arcene[5], and Handwritten Digits datasets were used.

- **Wine dataset:** This dataset is well studied and "well-behaved". It has 13 features, 3 classes and 178 samples. Hence a good problem for studying the differences in initialization and comparison of algorithms.
- **Digits dataset:** This dataset too has been studied extensively and has well documented behavior. It has 16 features, 10 classes and 1797 samples (10992 points).
- **Arcene dataset:** Arcene is a sparse dataset with 2 classes, 10000 features and 900 points which examine the effect of number of features used.

B. Particle Swarm Optimization

PSO was inspired by the social behavior of flocking of birds and originally developed by Eberhart and Kennedy in 1995 [7]. It is a population based stochastic optimization where the algorithm maintains a swarm of particles with each particle representing a solution to the optimization problem. PSO aims at finding the particle position that gives the best evaluation for a given fitness function.[2]

The following section describes the working of Particle Swarm Optimization and goes over PSO clustering and

hybrid PSO clustering algorithms [13]. For this purpose the following symbols are defined:

- N_d : dimension of data vector
- N_c : number of cluster centroids
- z_p : p^{th} data vector
- m_j : centroid vector of cluster j
- C_j : subset of datavector that form cluster j

One of the key components of clustering is the measure of similarity which is used for grouping data into predetermined number of clusters. Two prominent methods which are used to computer the similarity are the Euclidean distance, used for data vector clustering, and the cosine correlation measure, used for document clustering. Euclidean distance is used as similarity measure. Data vectors within a cluster are at a small 'Euclidean' distance from one another, and are associated with one centroid vector of that cluster. Distance of a vector to the centroid is determined using equation 1 :

$$d(z_p, m_j) = \sqrt{\sum_{k=1}^{N_d} (z_{pk} - m_{jk})^2} \tag{2}$$

Algorithm initially start with a set of randomly generated points where each point refers to the position of a particle in N_d dimensional space. Associated with each particle is its velocity vector. Each particle has the following information: x_i : The current position of the particle; v_i : The current velocity of the particle; y_i : The personal best position of the particle. A particle's position at the next time instance is then calculated as:

$$v_{i,k}(t+1) = w v_{i,k}(t) + c_1 r_{1,k}(t)(y_{i,k}(t) - x_{i,k}(t)) + c_2 r_{2,k}(t)(\hat{y}_k(t) - x_{i,k}(t)) \tag{3}$$

$$x_{i,k}(t+1) = x_{i,k}(t) + v_{i,k}(t+1) \tag{4}$$

where, w is the inertia weight, c_1 and c_2 are the acceleration constants, $r_{1,j}$(t),$r_{2,j}$(t) $\sim U(0,1)$ and k = 0,...,N_d. As is clear from equation 3, the velocity is updated based on three components: first is a fraction of its previous velocity, second is cognitive component which is a function of the distance of particle from its personal best position and third is social component which is a function of distance of particle from the global best position. The personal best position of a particle, defined to be the position which gives the best evaluation of the fitness function over all instances, is updated as:

$$y_i(t+1) = \begin{cases} y_i(t) \text{ if } f(x_i(t+1)) \geq f(y_i(t)) \\ x_i(t+1) \text{ if } f(x_i(t+1)) < f(y_i(t)) \end{cases} \tag{5}$$

C. Particle Swarm Optimization Clustering

In this context, every particle in PSO represents N_c cluster centroid vectors. Each particle is constructed as: x_i = (m_{i1},...,m_{ij},...m_{iN_c}) where m_{ij} is the cluster centroid of the i-th particle in cluster C_{ij}. The fitness function is defined in terms of the quantization error as:

$$J_e = \frac{\sum_{j=1}^{N_c} [\sum_{\forall z_p \in C_{ij}} d(z_p, m_j)/|C_{ij}|]}{N_c} \tag{6}$$

where d is the euclidean distance defined in equation 2 and $|C_{ij}|$ is the no. of data vectors belonging to the cluster C_{ij}.[13]

1) Algorithm for PSO clustering:

1) Number of particles = 10
2) Initialize each particle to have randomly selected N_c cluster centroids
3) For i in range t_{max}:
   a) For j in range No. of particles:
      i) For each data vector:
         A) Calculate the euclidean distance $d(z_p, m_{ij})$ to all cluster centroids C_{ij}
         B) Assign the data vector to the cluster such that the euclidean distance is minimum
      ii) Calculate the fitness function using equation 6
   b) Update local best position using equation 5
   c) Update the global best position as the position of particle which minimizes the fitness function
   d) Update the cluster centroids using equation 3,4

D. Hybrid-PSO Clustering

Hybrid-PSO algorithm is a hybrid of K-means and PSO methods of clustering. In this, K-means is executed once and the results of K-means are used to seed one of the particles in PSO clustering algorithm. Then PSO algorithm is executed.

1) Algorithm for Hybrid-PSO clustering:

1) Number of particles = 10
2) Execute K-means on the data and assign the calculated centroid to one particle
3) Initialize other nine particles to have randomly selected N_c cluster centroids
4) For i in range t_{max}:
   a) For j in range No. of particles:
      i) For each data vector:
         A) Calculate the euclidean distance $d(z_p, m_{ij})$ to all cluster centroids C_{ij}
         B) Assign the data vector to the cluster such that the euclidean distance is minimum
      ii) Calculate the fitness function using equation 6
   b) Update local best position using equation 5
   c) Update the global best position as the position of particle which minimizes the fitness function
   d) Update the cluster centroids using equation 3,4

III. Experimental Results

A. K-means: Random vs Refined Comparison

In this section the results of the silhouette scores from each of the databases will be discussed. In each test, the

random K-means method used was from the Scikit-learn[11] package for Python. This was also the basis for the random initialization within the refined method. Each clustering was limited to 50 iterations. The data subsets in the refined method were each 5% of the initial dataset.

The Wine set was tested by varying the number of subsets, J, within the refined method. A selection of these is shown in Figure 1 to demonstrate the saturation of the scoring as J increases. The number of clusters is also varied.

The results show the 2 methods are comparable at their peaks. This dataset is the least complex, in that it has a limited number of both features and classes. It is this lack of complexity which can account for the similar peaks. Each method as able to identify 3 distinct classes, which matches the true characteristics. However, when the number of clusters did not match the number of classes, the refined algorithm performed better for almost all values of J.

At 3 clusters, it should be noted that a small value of J caused the random method to perform better than the refined. This can be attributed to the added complexity in the refined model. If the data does not require the additional complexity, it may degrade overall performance.

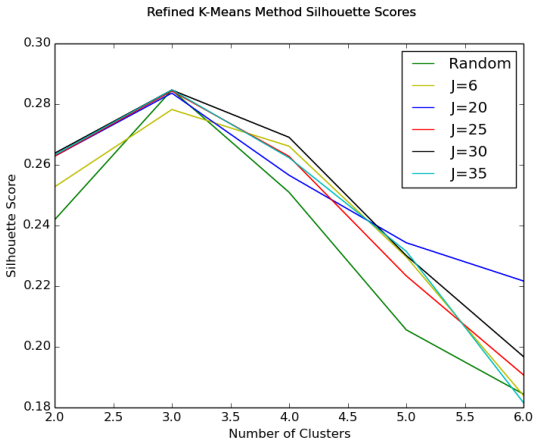


Fig. 1: Refined Subset results on Wine data

The same method of comparison as was used on the Wine set was used on the Digits set, with the exception of extending the range of clusterings. Since the Digits set has 10 classes, the range of tested clusters needed to be larger. The number of class labels the set has does not directly affect the algorithm, since these are unsupervised learning methods the training of the clusters does not use the labels. However, the structure of the data is more complex, with a comparable number of features as the Wine set.

The results of this set are shown in Figure 2. In this set, the refined method still shows improvement at higher J values, with saturation limiting the performance as it increases after that. The refined method identified 9 clusters as being optimal, with performance trailing off as k increased.

The random method peaked at 11 clusters, but it showed the smallest peak. The refined method with $J = 6$ was comparable, but it still outperformed random for most of the range. The saturated values of J outperform the random method across all k values.

This set showcases why the refined initialization method may be preferred to the random method. While there is more computation at the start of the training, it may yield better performance. This increased performance will show if the dataset is sufficiently complex, while simpler datasets may not see any improvement between the two methods. Both

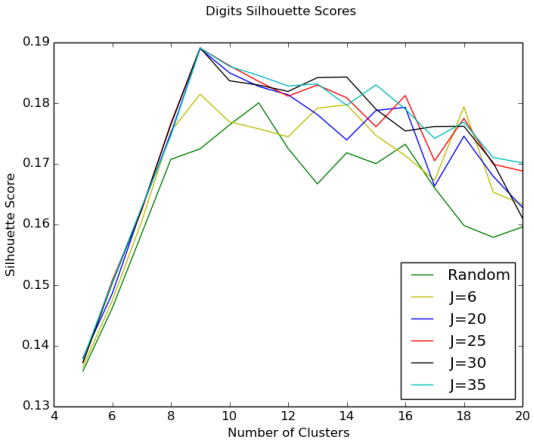


Fig. 2: Refined Subset results on Digits data

the Wine and the Digits data show the refined method will perform as well or better than the random method given a sufficient value of J. The last dataset, Arcene, examines the effects of feature selection on the methods. This set is sparse, with 10000 features and 2 classes. Running each method on the set showed a peak silhouette score at 2 clusters, with the refined method saturating at $J = 20$. The results of feature selection shown in Figure 3 are set at $k = 2$ and $J = 20$.

The graph shows the scoring as a function of the number of features. The feature selection was performed using recursive feature elimination on an L2 support vector machine[6]. One effect this can have on scoring is an increase in the silhouette score. The mathematical method of scoring uses the notion of distance. As features are eliminated, dimensions in the feature space are eliminated which causes smaller distances between points. However this will affect the random and refined methods equally, allowing comparison between them.

The methods have identical scores at 2 features. This greatly reduced feature space causes a simplified structure, leading to results similar in the Wine set. As the number of features increases, the refined method consistently performs better than the random method. The gap between the measures of performance is the greatest from 8 to 20 features. This region shows the refined method can maintain better cluster definition as complexity increases. There is a saturation point where the methods perform similarly, at 30+ features.

B. PSO based algorithms vs K-means

This section compares K-means with random initialization, PSO and hybrid PSO. The metrics used for comparison are:

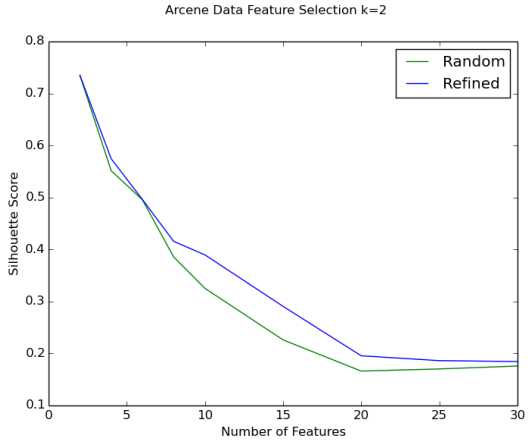


Fig. 3: Feature Selection results on Arcene data

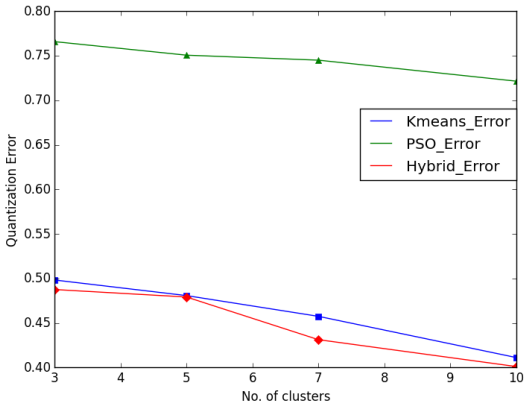


Fig. 4: Fitness function against number of clusters

- Quantization error
- Silhouette score

Besides the above metrics, the algorithms are also visualized and compared based on their plots of PCA reduced data [11].

The results are averaged over 10 simulation runs. All algorithms are run for 1000 iterations and 10 particles are used for PSO and hybrid PSO algorithms. The constants used for both of these PSO based algorithms are $w = 0.72$ and $c_1 = c_2 = 1.49$. These values are chosen as they deliver good convergence [13]. Wine and Digits data described above in II-A are used for comparison of the algorithms. Preprocessing is carried out on the datasets to standardize the features and generate comparable results.

TABLE I: Algorithm Comparison

Dataset	Algorithm	Quantization Error	Silhouette score
Wine	K-means	0.4987	0.3008
	PSO	0.7964	0.1402
	PSO-Hybrid	0.4779	0.3297
Digits	K-means	66.55	0.1457
	PSO	17.58	0.0308
	PSO-Hybrid	15.62	0.0288

Table I lists the performance of the three algorithms on Wine and Digits dataset averaged over 10 simulations.

One thing to be noted here is that although the Quantization error is comparable for the algorithms for a given dataset, it is not comparable between different datasets. This is because the quantization error depends on the number of clusters, the preprocessing of data, number of samples among other factors which vary greatly across datasets.

For the wine dataset it is clear that PSO performs worse than K-means when compared over the quantization error as well as the silhouette score. However a significant improvement can be seen in the PSO-Hybrid algorithm. When one particle in PSO algorithm is seeded with the results from the K-means algorithm, the resulting algorithm performs much better than the original PSO and marginally better than the standard random K-means.

In the case of digits dataset it is observed that PSO and Hybrid PSO have much lower quantization error compared to K-means with the Hybrid-PSO having the least error of all three. However K-means algorithm gives us a little better clustering than the two PSO based algorithms. It can be seen from the above results that in general there is an improvement in the performance when PSO is seeded with the K-means results.

Figure 5 shows the plots of the three algorithms on PCA reduced Wine and digits datasets. The datasets have been reduced to two most significant features to assist 2-D visualization. The results in Table I corroborate what is seen here in Figure 5.

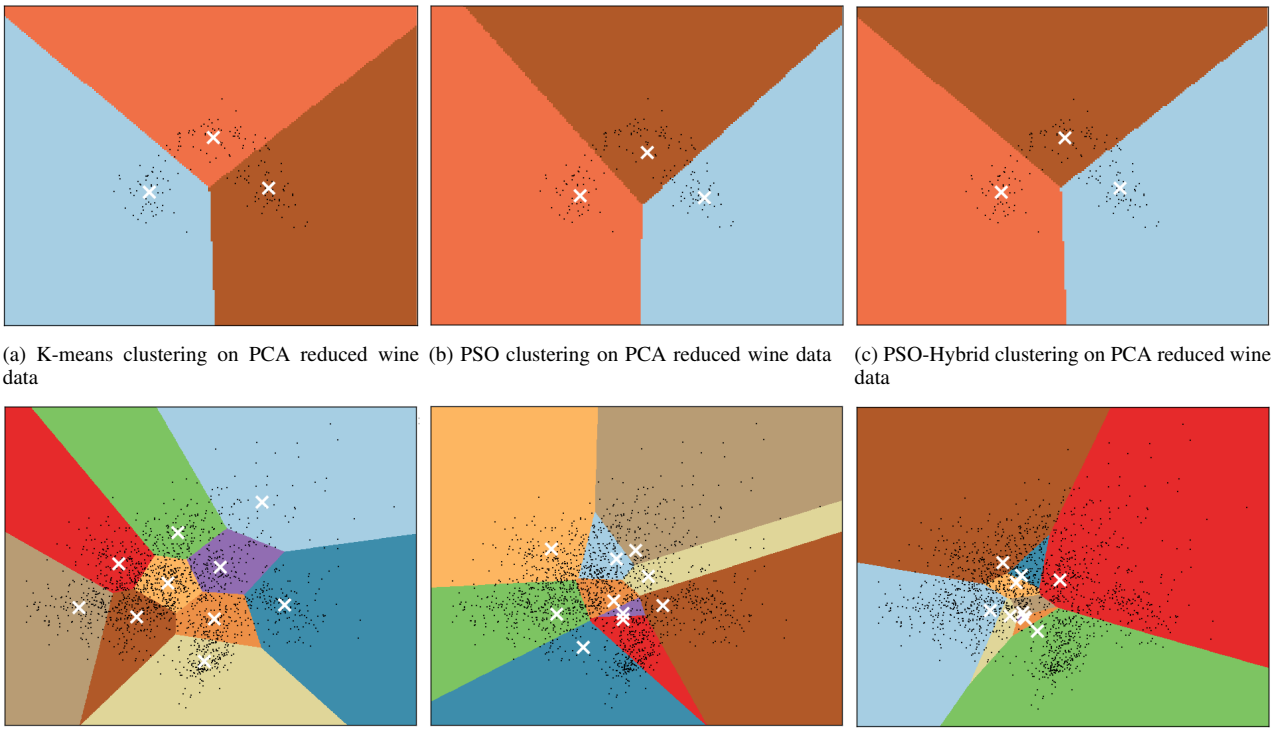
For the wine dataset it is observed that the Hybrid PSO and K-means, which have higher silhouette score, exhibit a better plot than PSO which has a lower silhouette score. For the digits dataset, the K-means algorithm has the best silhouette score and both the PSO based algorithms feature near the middle of the silhouette score scale. Hence the plot of the K-means for the digits shows much better distribution than the other two.

Figure 4 shows the effect of increasing the number of clusters for the three algorithms over wine dataset. From equation 6 it is clear that an increase in the number of clusters, N_c, will result in decrease of the quantization error. This is illustrated in Figure 4. It can also be observed from the figure that Hybrid PSO performs better than the other two algorithm consistently as the number of clusters are increased.

IV. Conclusion

As a technique, the random initialization K-means is a fast and simple method for examining data structures. However it does have faults, with opportunities to improve. This paper has demonstrated techniques to improve the performance of K-means through refined initial centers and particle swarm optimization.

Both improvements are capable of creating less distortion while clustering data. The key to unsupervised learning



(a) K-means clustering on PCA reduced wine data (b) PSO clustering on PCA reduced wine data (c) PSO-Hybrid clustering on PCA reduced wine data

(d) K-means clustering on PCA reduced digits data (e) PSO clustering on PCA reduced digits data (f) PSO-Hybrid clustering on PCA reduced wine data

Fig. 5: Clustering for K-means, PSO and Hybrid PSO on PCA reduced dataset

performance is understanding how and when each of these should be used. There is no one technique which will always lead to the best clustering in a dataset.

In future, determination of the number of clusters dynamically using Silhouette score can be incorporated. The studies of PSO-Hybrid algorithm can be extended to cover more variants of datatypes such as document clustering and image clustering.

V. Contribution

In this paper Ben Lickiss examined the refined vs random K-means comparison. Paresh Bhambhani examined K-means vs PSO. Both contributed to the Introductory and Concluding material.

References

[1] Paul S Bradley and Usama M Fayyad. Refining Iintial Points for K-Means Clustering. *Proceedings of the 15th International Conference on Machine Learning (ICML98)*, pages 91–99, 1998.

[2] Edwin KP Chong and Stanislaw H Zak. *An introduction to optimization*, volume 76. John Wiley & Sons, 2013.

[3] Sandra Cohen and Leandro N de Castro. Data clustering with particle swarms. In *Evolutionary Computation, 2006. CEC 2006. IEEE Congress on*, pages 1792–1798. IEEE, 2006.

[4] Edward Forgy. Cluster analysis of multivariate data: efficiency vs interpretability of classifications. *Biometrics*, 21:768–769, 1965.

[5] Isabelle Guyon, Steve R. Gunn, Asa Ben-Hur, and Gideon Dror. Result analysis of the nips 2003 feature selection challenge, 2004.

[6] Isabelle Guyon, Jason Weston, Stephen Barnhill, and Vladimir Vapnik. Gene selection for cancer classification using support vector machines. *Machine Learning*, 46(1-3):389–422, 2002.

[7] James Kennedy, James F Kennedy, Russell C Eberhart, and Yuhui Shi. *Swarm intelligence*. Morgan Kaufmann, 2001.

[8] M. Lichman. UCI machine learning repository, 2013.

[9] J MacQueen. Some methods for classification and analysis of multivariate observations. *Proceedings of the Fifth Berkeley Symposium on Mathematical Statistics and Probability*, 1:281–297, 1967.

[10] M Omran, Ayed Salman, and Andries P Engelbrecht. Image classification using particle swarm optimization. In *Proceedings of the 4th Asia-Pacific conference on simulated evolution and learning*, volume 1, pages 18–22. Singapore, 2002.

[11] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, and P. Prettenhofer. Scikit-learn: Machine learning in Python.

[12] Peter J. Rousseeuw. Silhouettes: A graphical aid to the interpretation and validation of cluster analysis. *Journal of Computational and Applied Mathematics*, 20:53–65, 1987.

[13] DW Van der Merwe and Andries P Engelbrecht. Data clustering using particle swarm optimization. In *Evolutionary Computation, 2003. CEC'03. The 2003 Congress on*, volume 1, pages 215–220. IEEE, 2003.