Scalable Kernel k-Means via Centroid Approximation

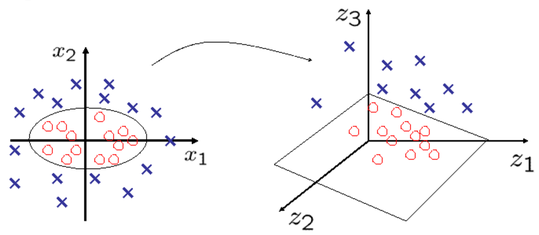
Byung Kang (byungkon@kaist.ac.kr), Woosang Lim (quasar17@kaist.ac.kr), Kyomin Jung (kyomin@kaist.edu)

Abstract

- We present a novel algorithm based on scalable centroid approximation that accelerates kernel k-means down to a sub-quadratic per-iteration complexity of $O(n^{1+\delta})$ for any $\delta \in (0, 1)$.
- We prove that our algorithm's approximation of the distortion is within a factor of $(1+n^{-\delta})$ of the kernel k-means algorithm's distortion, and show the effectiveness of our algorithm through extensive experiments.

1. Introduction

k-means clustering is a popular iterative clustering algorithm for Euclidean data. For non-Euclidean data or data with non-linear separability, the kernel method is often applied to the clustering algorithm.



However, kernelized clustering algorithms suffer high per-iteration time and space complexity of $\Theta(n^2)$.

In this work, we present a novel approximation algorithm called CATS (**C**entroid **A**pproximation **T**hrough **S**ampling) that accelerates kernel k-means to a time complexity of $O(n^{1+\delta})$ while achieving high accuracy.

In terms of memory usage, our algorithm's storage requirement is $O(n\ell)$, where ℓ is the number of samples we use. On the other hand, spectral clustering algorithms, such as NJW [3], require $\Theta(n^2)$ elements, due to affinity matrix computation.

2. Kernel k-Means

Given a kernel function $\kappa(x, y) = \phi(x) \cdot \phi(y)$, the kernel k-means over n points proceeds as follows:

1. Initialize partition $\bar{C}$
2. For each cluster $C \in \bar{C}$, compute and cache centroid $m_C \cdot m_C$
3. Assign point x_i to cluster
$$\operatorname*{argmin}_C \left\{ \kappa(i,i) - \frac{2}{|C|} \sum_{j \in C} \kappa(i,j) + m_C \cdot m_C \right\}$$
4. If not converged, goto line 2

The computation of $m_C \cdot m_C = \frac{1}{|C|^2} \sum_{i,j \in C} \kappa(i,j)$ costs $\Theta(n^2)$ operations per iteration.

3. Algorithm

Centroid Approximation: Instead of computing the exact centroid m_C, we compute the *approximate centroid* $\tilde{m}_C$ for each cluster C. We randomly sample ℓ points into the set $S \subseteq C$, and define the approximate centroid as:

$$\tilde{\mathbf{m}}(\boldsymbol{\alpha})_C = \sum_{i \in S} \alpha_i \phi(\mathbf{x_i}),$$

where $\boldsymbol{\alpha} = (\alpha_1, \cdots, \alpha_\ell)$ are the coefficients to be optimized. The $\boldsymbol{\alpha}$ is optimized w.r.t. the *Distortion*:

$$D_C(\boldsymbol{\alpha}) = \sum_{i \in C} \|\phi(x_i) - \tilde{m}(\boldsymbol{\alpha})_C\|^2 \qquad (1)$$

By differentiating Eqn 1, we get:

$$\boldsymbol{\alpha} = \frac{M^+ L \mathbf{e_1}}{n}, \qquad (2)$$

where $L \in \mathbb{R}^{\ell \times n}$ is a matrix that contains the kernel values between points in S and C, and $M \in \mathbb{R}^{\ell \times \ell}$ is a matrix containing the kernel values over $S \times S$, $\mathbf{e_1}$ is a vector of 1s, and M^+ is the pseudoinverse of M.

Inspiration: Johnson-Lindenstrauss lemma [4].

- Low-dimensional embedding approximately preserves pairwise distances, under certain conditions.
- Projecting centroid onto sampled subspace might yield good approximation.

Stopping Criterion: This approximation leads to the possibility of oscilation towards the end of the iteration. To overcome this problem, we halt the iteration if the variance of the past w distortions is below a given small threshold θ.

4. Performance Analysis

Let $S \subseteq C$ be a uniformly sampled subset, for which $|S| = n^\delta$, for any $\delta \in [0, \frac{1}{2}]$. Then,

Theorem 1 *The approximated distortion is within* $O\left(1 + \frac{1}{n^\delta}\right)$ *of the true distortion.*

Theorem 2 *The per-iteration time complexity is* $O(n^{1+\delta})$.

i.e. We have a near-optimal guarantee on approximating the true distortion, with sub-quadratic time complexity.

5. Experiments

We tested our algorithm on the UCI database [2] and the MNIST database [1]. We compare CATS using our stopping criterion and kernel k-means (KKM) run until convergence ($\ell = \sqrt{n/k}$):

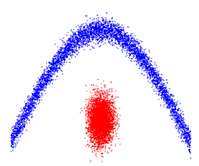
		MNIST	UCI
Distortion	KKM	19.1273	3.5904
	CATS	19.134 (10^{-4})	3.5904 (10^{-7})
Time (sec.)	KKM	1854	1253
	CATS	110 (47)	76 (4.8)
NMI		0.91 (0.019)	0.99 (10^{-4})

Average time and distortion over 100 trials, with standard deviation in parentheses.

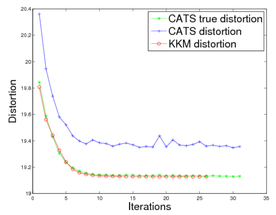
Comparison against NJW spectral clustering [3]:

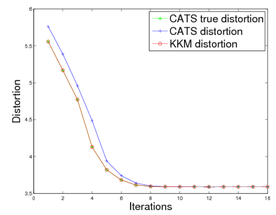
	NJW	KKM	CATS
Time (sec.)	180	76	12
True NMI	0.553	0.527	0.530

For qualitative evaluation, we also tested CATS on the following two synthetic datasets (k=5 and 2, respectively):



The plot of distortion per each iteration shows that CATS matches KKM well.





We also report the results of varying ℓ:

	ℓ= 8	16	33	66
Distortion	0.34 (0.004)	0.33 (0.004)	0.33 (10^{-5})	0.34 (10^{-7})
Time (sec.)	65 (8)	71 (4)	96 (3)	233 (17)
NMI	0.92 (0.04)	0.94 (0.04)	0.99 (0.0002)	0.99 (0.0005)
True NMI	0.64	0.65	0.63	0.64

References

[1] Y. LeCun, C. Cortes. MNIST Database. http://yann.lecun.com/exdb/mnist/
[2] A. Frank, A. Asuncion. UCI Machine Learning Repository. http://archive.ics.uci.edu/ml
[3] A. Ng, M. Jordan, Y. Weiss. On Spectral Clustering: Analysis and an Algorithm. In *NIPS '01*
[4] W. Johnson, J. Lindenstrauss. Extensions of Lipschitz mappings into a Hilbert space. In *Contemporary Mathematics*, 1984.