KMEANS and Kernel KMEANS

A comparative study of classical and kernel kmeans for pixel clustering.

Ezukwoke K.I[1], Zareian S.J[2]

[1, 2] Department of Computer Science
Machine Learning and Data Mining
{ifeanyi.ezukwoke, samaneh.zareian.jahromi}@etu.univ-st-etienne.fr
University Jean Monnet, Saint-Etienne, France

Abstract

Kmeans is a simple and efficient clustering algorithm. In this paper we study kernel Kmeans and compare it with classical Kmeans. We take an experimental analysis on image dataset for pixel clustering. We briefly introduce multiple kernel learning and its applications in kernel kmeans. We conclude by expanding on the performance of both algorithms to see which is most suited for pixel clustering.

Keywords
Partition clustering, K-Means, kernel K-Means.

1 INTRODUCTION

K-Means clustering is a fast, robust, and simple algorithm that gives reliable results when data sets are distinct or well separated from each other in a linear fashion. It is best used when the number of cluster centers, is specified due to a well-defined list of types shown in the data. However, it is important to keep in mind that K-Means clustering may not perform well if it contains heavily overlapping data, if the Euclidean distance does not measure the underlying factors well, or if the data is noisy or full of outliers.

2 PARTITION CLUSTERING

Clustering is the task of gathering samples into groups of similar samples according to some predefined similarity and they find applications in *data compression*, *data summurization for recommender systems*, *similarity grouping of web search result, stock indices and customer profiles.*
KMeans falls under the clustering category called **partition clustering**. This approach used a technique of splitting the datasets into subgroups of $k - clusters$ and iteratively tries to find the best $k - cluster$ that best explains the partition of a data.

2.1 KMEANS

The objective of traditional clustering methods is to partition training vectors by using similarity criteria applied on the basis of Euclidean metrics. More precisely, the Euclidean distance or inner product is used to measure the similarity between the training vectors in the original vector space, $\{x_i, i = 1, \ldots, N\}$. The objective function we try to minimize in KMeans is given by

$$\arg\min_{w_k} \left\{ \sum_{k=1} K \sum_{X_t \in C_k} \|x_t - \mu_k\|^2 \right\} \quad (1)$$

Where μ_k denotes the *kth* cluster's centroid. In an optimal K-means solution, the centroid, say μ_k is associated with a training vector x_j that yields the minimum distance among all the centroids.

Algorithm 1: Classical K-Means clustering algorithm

Input : x
Output : $x_k \in C_k$
begin
  while *not converged* **do**
    Randomly initialize μ cluster center;
    Compute the distance of each point in x from center μ $\arg\min \sum_{c=1}^{k} \sum_{x_i \in C_k} ||x_i - m_c||^2$
  end
  return x_k
end

2.2 Kernel KMEANS

Kernel Kmeans is a non-lnear version of the classical kmeans algorithm. The LSP theorem assures that K-means have a kernelized formulation.[1]

T H E O R E M 1. (Learning subspace property(LSP))
Consider an optimizer aiming at finding

$$\underset{\{w_k \in \mathbb{R}^M, \forall k\}}{\arg\min} \ \varepsilon \left(w_k^T x_j, \forall k, j; || \sum_{k=1}^{K} \beta_k^{(l)} w_k ||^2, \forall l \right) \tag{2}$$

subject to the equality constraints

$$c_p(w_k^T x_j, \forall k, j) = 0, \ p = 1, \ldots, P, \tag{3}$$

and inequality constraints

$$d_q \left(w_k^T x_j, \forall k, j; || \sum_{k=1}^{K} \beta_k^{(l)} w_k ||^2, \forall l \right) \leq 0, \tag{4}$$

$$q = 1, \ldots, Q.$$

Here $\forall k, \forall j$, and $\forall l$ stand for $k = 1, \ldots, K, j = 1, \ldots, N$,and $l = 1, \ldots, L$ respectively. Suppose further that $\varepsilon(.)$ and $\{d_q(.), q = 1, \ldots, Q\}$ are all monotonically increasing functions w.r.t. $|| \sum_{k=1}^{K} \beta_k^{(l)} w_k ||$ for $l = 1, \ldots, L$. Then, under the given constraints,

$$\underset{w_k \in \mathbb{R}^M, \forall k}{\arg\min} \ \varepsilon = \underset{w_k = X a_k, a_k \in \mathbb{R}^N, \forall k}{\arg\min} \ \varepsilon.$$

This equivalence guarantees an equally good result even if the solution is restricted to the learning subspace property (LSP):

$$w_k \in span[X], \ \textit{equivalently} \ , w_k = X a_k,$$

$$\textit{for some } q_k \in \mathbb{R}^N, \forall k.$$

The LSP holds for the K-means learning model, since it is based on an $l2 - norm$ clustering criterion that satisfies Theorem 1. According to the theory, the LSP condition holds for the K-means learning model.

In kernel methods, the conventional Euclidean inner product can be extended to a versatile nonlinear inner product, represented by a Mercer kernel $K(x, y)$ of two vectors x and y. Such a generalized inner product will be much more amenable to complex and big data analysis. Let X be the original space, which can be assumed to be a subset of R . In functional analysis, a Mercer kernel is a function $k(\Delta, \Delta)$ from $X \times X$ to R that satisfies the Mercer condition [2, 3, 4, 5].

The Mercer condition
Let $K(x, y)$ be a continuous symmetric kernel that is defined in a closed interval for x and y. The function $K(x, y)$ is called a Mercer kernel if it meets the Mercer condition that

$$\int K(x, y) h(x) h(y) dx dy \geq 0, \tag{5}$$

for any squarely integrable function $h(x)$, i.e. $\int h(x)^2 dx$ is finite. Mercer's theorem[177] states that there exists a reproducing kernel Hilbert space H and a mapping

$$\phi : x \to \phi(x), \phi(x) \in H,$$

such that the inner product for H is represented by

$$k(x, y), i.e. k(x, y) = \phi(x)^T \phi(y).$$

T H E O R E M 2.1 (Mercer's theorem)
The Mercer condition is necessary and sufficient for $K(x, y)$ to be expandable into an absolutely and uniformly convergent series

$$K(x,y) = \sum_{i=1}^{\infty} c_i \tilde{q_i}(x)\tilde{q_i}(y), \qquad (6)$$

with positive coefficients $\{c_i > 0\}$, i.e. it may be factorized as

$$K(x,y) = \vec{q}(x)^T \vec{q}(y),$$

where we denote

$$\vec{q}(x) \equiv [\sqrt{c_1}\tilde{q_1}(x)\sqrt{c_2}\tilde{q_2}(x)\ldots]^T.$$

The Mercer condition assures that there is a Hilbert vector space, endowed with a Euclidean-like distance metric – a property that is vital for many learning models [2, 3, 6].

THEOREM 2.2 (Mercer kernels)
This theorem formally establishes that the Mercer condition holds for some important kernel functions, qualifying them as Mercer kernels.
(a) Denote a kernel function as

$$K(x,y) = \sum_{i=1}^{J} c_i \tilde{q_i}(x)\tilde{q_i}(y), \ \textit{where} \qquad (7)$$

$$q_i(x) = \prod_{j=1}^{M} x_j^{r_j},$$

where $\{r_j, \forall j\}$ are natural numbers, is expressed as functions of the elements of the vector x. The function $K(x, y)$ is a Mercer kernel if and only if all its defining coefficients $\{ci, \forall i\}$ are non-negative.
(b) A typical polynomial kernel function,

$$K(x,y) = (1 + \frac{x.y}{\sigma^2})^p, \qquad (8)$$

where p is a positive integer, is always a Mercer kernel.
(c) A Gaussian RBF kernel function,

$$K(x,y) = \exp\left\{-\frac{||x-y||^2}{2\sigma^2}\right\}, \qquad (9)$$

is always a Mercer kernel. [1] So in this algorithm the data x_i is projected into a higher dimensional subspace $\phi(x_i)$ where we are only concerned in the dot products of the feature vectors.

Algorithm 2: Kernel K-Means clustering algorithm

Input : x
Output : $x_k \in C_k$

```
begin
    while not converged do
        Randomly initialize μ cluster center;
        Compute the distance of each point in x from center μ
        arg min Σ_{c=1}^{k} Σ_{x_i∈C_k} ||φ(x_i) − m_c||^2
    end
    return x_k
end
```

2.3 Kernels

We introduce the commonly used kernels and a brief overview of Multiple kernels used.

- **Linear kernel**

$$\kappa(x_i, x_j) = \mathbf{x}_i \mathbf{x}_j^T \qquad (10)$$

- **Polynomial kernel**

$$\kappa(x_i, x_j) = (\mathbf{x}_i \mathbf{x}_j^T + c)^d \qquad (11)$$

where $c \geq 0$ and d is the degree of the polynomial usually greater than 2.

- **RBF(Radial Basis Function) kernel**
Sometimes referred to as the **Guassian kernel**.

$$\kappa(x_i, x_j) = \exp(-\gamma||\mathbf{x}_i - \mathbf{x}_j||^2) \qquad (12)$$

where $\gamma = \frac{1}{2\sigma^2}$.

- **Sigmoid kernel**

$$\kappa(x_i, x_j) = \tanh(\gamma \mathbf{x}_i \mathbf{x}_j^T + c) \qquad (13)$$

where $c \geq 0$ and $\gamma = \frac{1}{2\sigma^2}$.

- **Cosine kernel**

$$\kappa(x_i, x_j) = \frac{\mathbf{x}_i \mathbf{x}_j^T}{||\mathbf{x}_i|| ||\mathbf{x}_j||} \tag{14}$$

2.4 Muli-kernels

The reason behind the use of multiple kernels is similar to the notion of multiclassification, where cross-validation is used to select the best performing classifier [**?**]. By using multiple kernels, we hope to learn a different similarity uncovered using single kernels.

We can prove from **Mercer's Theorem** that a kernel is **Positive SemiDefinite (PSD)** if $\kappa(x_i, x_j) \geq 0$. Hence by performing arithmetic or any mathematical operation on two or more kernel matrix, we obtain a new kernel capable of exploiting different property or similarities of training data.

Given a kernel κ, we prove that κ is PD if

$$\langle u, \kappa u \rangle \geq 0 \tag{15}$$

Proposition: *A symmetric function* κ*:* $\times$ $\chi \to \mathbb{R}$ *is positive semidefinite* $\iff$ κ
Proof:
Suppose that κ is a kernel which is the inner product of the mapping functions $\langle \phi(x_i)\phi(x_j) \rangle$. κ is a kernel if its inner product are positive and the solution of $\kappa u = \lambda u$ gives non-negative eigenvalues.

So that,

$$\langle u, \kappa u \rangle = \sum_{i=1}^{N} u_i (\kappa u)_i \tag{16}$$

$$= \sum_{i=1}^{N} u_i \sum_{j=1}^{N} \langle \phi(x_i)\phi(x_j)_{\mathcal{H}} \rangle \, u_j \tag{17}$$

Where $\mathcal{H}$ represents the Hilbert space we project the kernel in[7].

$$= \left\langle \sum_{i=1}^{N} u_i \phi(x_i), u_j \phi(x_j) \right\rangle_{\mathcal{H}} \tag{18}$$

$$\langle u, \kappa u \rangle = \left\| \sum_{i=1}^{N} u_i \phi(x_i) \right\|_{\mathcal{H}}^2 \geq 0 \tag{19}$$

Therefore κ is positive definite.

Using this property of the kernel κ we are able to create serveral other kernels including

- **LinearRBF**
  Here we combine two kernels, precisely **Linear and RBF kernel** using their inner product.

$$\hat{\mathbf{K}}_{\mathbf{linrbf}} = \kappa(x_i, x_j)^T \kappa(x_i, x_l) \tag{20}$$

- **RBFPoly**
  Here we combine **RBF and Polynomial kernels** using their inner product.

$$\hat{\mathbf{K}}_{\mathbf{rbfpoly}} = \kappa(x_i, x_j)^T \kappa(x_i, x_l) \tag{21}$$

- **RBFCosine**
  RBF and Cosine kernels using their inner product.

$$\hat{\mathbf{K}}_{\mathbf{rbfcosine}} = \kappa(x_i, x_j)^T \kappa(x_i, x_l) \tag{22}$$

- **EtaKernel**
  The **EtaKernel** is a composite combination of **LinearRBF, RBFPoly and RBFCosine** and it is given by

$$\hat{\mathbf{K}}_{\mathbf{etarbf}} = \hat{\mathbf{K}}_{\mathbf{linrbf}}^T \hat{\mathbf{K}}_{\mathbf{rbfpoly}} + \hat{\mathbf{K}}_{\mathbf{rbfpoly}}^T \hat{\mathbf{K}}_{\mathbf{rbfcosine}}$$

3 EXPERIMENTAL RESULT

We experiment on images dataset benchmark.

3.1 Dataset

3.2 Performance analysis

4 CONCLUSION

References

[1] Kung,S.Y.(2014).*Kernel Methods and Machine Learning.*Cambridge University Press; p: 11-48.

[2] R. Courant and D. Hilbert. *Methods of Mathematical Physics*. New York: Interscience, 1953.

[3] J. Mercer. *Functions of positive and negative type and their connection with the theory of integral equations.* Phil. Trans. Royal Soc., A209:415–446, 1909.

[4] V. N. Vapnik. *The Nature of Statistical Learning Theory.* New York: Springer-Verlag, 1995.

[5] V. N. Vapnik. *Statistical Learning Theory.* New York: Wiley, 1998.

[6] R. Courant and D. Hilbert. *Methods of Mathematical Physics, volumes I and II.* New York: Wiley Interscience, 1970.

[7] John Shawt-Taylor, Nello Cristianini. *Kernel Methods for Pattern Analysis.* Cambridge University Press, ISBN:9780511809682, pg47-83, 2011.

[8] Pearson K. *On lineas and Planes of closest fit to systems of points in space. Philos Mag A*, 6:559–572, 1901.

[9] Zhu J, Hastie T. *Kernel logistic regression and import vector machine. J Comput Graphic Stat*, 14:185–205, 2005.

[10] Saunders C., Gammerman A., Vovk V., *Ridge Regression Learning Algorithm in Dual Variables. ICML 15th Internationl Conforence on Machine Learning.*, 1998.