Proceedings of Machine Learning Research 95:343-358, 2018 ACML 2018

A Faster Sampling Algorithm for Spherical k-means

Rameshwar Pratap RAMESHWAR.PRATAP@GMAIL.COM

Anup Deshmukh* DESHMUKH.ANAND@IIITB.ORG
IIIT Bangalore

Pratheeksha Nair* PRATHEEKSHA.NAIR@IIITB.ORG
IIIT Bangalore

Tarun Dutt* TARUN.DUTT@IIITB.ORG
IIIT Bangalore

Abstract

The *Spherical k-means* algorithm proposed by (Dhillon and Modha, 2001) is a popular algorithm for clustering high dimensional datasets. Although their algorithm is simple and easy to implement, a drawback of the same is that it doesn't provide any provable guarantee on the clustering result. (Endo and Miyamoto, 2015) suggest an adaptive sampling based algorithm (*Spherical k-means++*) which gives near optimal results, with high probability. However, their algorithm requires k sequential passes over the entire dataset, which may not be feasible when the dataset and/or the values of k are large. In this work, we propose a Markov chain based sampling algorithm that takes only one pass over the data, and gives close to optimal clustering similar to *Spherical k-means++*, *i.e.*, a faster algorithm while maintaining almost the same approximation. We present a theoretical analysis of the algorithm, and complement it with rigorous experiments on real-world datasets. Our proposed algorithm is simple and easy to implement, and can be easily adopted in practice.

Keywords: Spherical k-means; Unsupervised learning; Collaborative filtering; Document clustering; Markov chain.

1. Introduction

The *Spherical k-means* is an important problem in unsupervised learning. It is similar to the k-means clustering problem and uses *cosine similarity* as a similarity/distance measure instead of *euclidean distance*. (Kleinberg et al., 1998) showed that the *Spherical k-means* problem is NP-hard. (Dhillon and Modha, 2001) proposed an algorithm (SPKM) for the problem which consist of two steps. In the first step, k data points are sampled uniformly at random from the given set of data points, and it is called as the *seeding step*. In the second step, the solution is refined iteratively based on the cosine similarity between cluster centers and the points belonging to that cluster. The second step is repeated until there is little or no improvement from the previous iteration. Other than the similarity measure, the second step is similar to *Llyods-iterations* (Lloyd, 2006) for the k-means problem .

* Equal contribution.

1.1. Initial seeding for Spherical k-mean problem

The task of locating k initial cluster centers is important in obtaining high quality clustering. Although the SPKM (Dhillon and Modha, 2001) algorithm is simple and efficient, it depends on the initial values of the k cluster centers. As mentioned above, the initial k data points are selected uniformly at random from the dataset. This may lead to arbitrarily poor clustering performance when the distribution of points across the underlying optimal clusters is non-uniform.

The *Spherical k-means++* (SPKM++) algorithm, motivated by *k-means++*(Arthur and Vassilvitskii, 2007), suggests an adaptive sampling strategy to sample k initial points. Considering them as cluster centers, they give a solution that is $O(\log k)$-competitive with the optimal solution, without making any assumptions on the data. However, this approach requires k sequential passes over the data making it impractical even for moderate values of k, and/or when the dataset is large.

1.2. Our contribution

Contributions of the paper are two-fold and we summarize them as follows:

- We conducted thorough experimental evaluations of SPKM++ (Endo and Miyamoto, 2015) on publicly available datasets to demonstrate its applicability, which was not addressed in their paper.We obtain improved clustering quality in addition to better running time with respect to *vanilla* SPKM (Dhillon and Modha, 2001).
- We propose a Markov chain based algorithm (SPKM-MCMC) for initial seeding of k points. As opposed to k passes required by SPKM++, our algorithm requires only one pass over the data for the initial seeding. The theoretical guarantee on the clustering cost of SPKM-MCMC algorithm is close to SPKM++ while simultaneously achieving a significant speed-up in the seeding time [1]. We complement the theoretical analysis of our result by rigorous experiments on publicly available datasets.

Our results are presented below:

Theorem 1 *Let $\mathcal{X}$ be a set of n vectors in d-dimensional unit sphere, $\epsilon \in (0,1)$, k be a positive integer, and $\mathbf{C}$ be an output of Algorithm 1 consisting of initially sampled k seeding points, and $m = 1 + \frac{4}{\epsilon}\log\frac{4k}{\epsilon}$. Then,*

$$\mathbb{E}[\Phi_{\mathbf{C}}(\mathcal{X})] \leq 4(\log k + 2)\Phi_{\mathrm{OPT}}(\mathcal{X}) + \epsilon \mathrm{AV}(\mathcal{X}),$$

where $\Phi_{\mathbf{C}}(\mathcal{X})$ denotes the clustering cost of the algorithm; $\Phi_{\mathrm{OPT}}(\mathcal{X})$ denotes the cost of underlying optimal clustering; $\mathrm{AV}(\mathcal{X}) = |\mathcal{X}| - \sum_{\mathbf{x}\in\mathcal{X}}\langle \mathbf{x}, \mathrm{FM}(\mathcal{X})\rangle$; and $\mathrm{FM}(\mathcal{X})$ denotes the Fisher-Mean of the set of vectors $\mathcal{X}$. The seeding time of the algorithm is $O(nd + \frac{1}{\epsilon}k^2 d\log\frac{k}{\epsilon})$.

Remark 1 *There are two terms in the upper bound on the expected clustering cost – the first term is $4(\log k + 2)\Phi_{\mathrm{OPT}}(\mathcal{X})$ which is same as the clustering cost of SPKM++. The second term $\epsilon\mathrm{AV}(\mathcal{X})$ is an additive error due to the Markov chain approximation. The term $\mathrm{AV}(\mathcal{X})$ is the expected cost of the* SPKM *when $k = 1$, and the cluster center is sampled uniformly from $\mathcal{X}$.*

1. Seeding time is the time required to sample k initial vectors.

Remark 2 *We compare and contrast the effect of chain length on the clustering time and the clustering cost. As $m = 1 + \frac{4}{\epsilon}\log\frac{4k}{\epsilon}$, for a fixed value of k, a larger chain length leads to a smaller value of ϵ. A smaller value of ϵ renders the clustering cost of our algorithm to that of SPKM++. Further, as ϵ tends to zero, the seeding time tends to infinity. Thus, the clustering cost is inversely proportional to the chain length, while the clustering time is proportional to the chain length. However, we experimentally validate that even a small chain length gives good clustering performance – smaller clustering time as well as cost.*

Remark 3 *We compare the seeding time of SPKM++ and our algorithm. The seeding time of SPKM++ is $O(ndk)$, while the seeding time of our algorithm is $O\left(nd + \frac{1}{\epsilon}k^2 d\log\frac{k}{\epsilon}\right)$. We obtain a higher speedup with respect to SPKM++ for a larger value of k. This also reflected in our experimental work mentioned in Subsection 5.2.*

1.3. Comparison of k-means and Spherical k-mean:

The *k-means* (KM) clustering algorithm minimises the distance between cluster centers and member data points. On projecting these points onto a unit sphere, the sum of squared distances from the cluster centers acts as a natural measure of dispersion and is the motivation behind spherical k-means (SPKM) (Hill et al., 2013). Since the document vectors are normalised and have unit L^2 norm (Dhillon and Modha, 2001), the Euclidean distance between two vectors is monotonic to their cosine similarity. This may lead to confusion that both KM and SPKM give the same clustering results. However, we emphasize that this is not the case. The boundary between two clusters in the SPKM is a hyperplane passing through the origin. Such a hyperplane produces a hypercircle on intersection with a unit sphere which is then used as the measure of closeness. Thus, the SPKM partitions the unit sphere using a collection of great hypercircles. On the other hand, the boundary between clusters in KM (hyperplane) does not generally intersect with the unit sphere to produce a great hypercircle. Section 3.6 of (Dhillon and Modha, 2001) gives a detailed discussion.

Organization of the paper: In Section 2, we present the necessary background that is required to understand the paper. In Section 3, we present our algorithm – Markov chain based concept decomposition of text documents, and its analysis. In Section 4, we discuss some fundamental applications of our result. In Section 5, we complement our theoretical results with extensive experimentation on real-world datasets. Finally, in Section 6, we conclude our discussion.

2. Background

We start with a word of notations. Let $||\mathbf{x}||$ denote L^2 norm of the vector $\mathbf{x}$. *Inner product* between two d-dimensional vectors $\mathbf{x}$ and $\mathbf{c}$ is denoted as $\langle \mathbf{x}, \mathbf{c}\rangle$ and is defined as $\langle \mathbf{x}, \mathbf{c}\rangle = \mathbf{x}^T\mathbf{c} = \sum_{i=1}^{d} x_i c_i$. *Angular dissimilarity* between two unit vectors $\mathbf{x}$ and $\mathbf{c}$ is defined as $\mathrm{d}^{(1)}(\mathbf{x}, \mathbf{c}) = 1 - \langle \mathbf{x}, \mathbf{c}\rangle$. Further angular dissimilarity between a vector $\mathbf{x}$ and a set of unit vectors $\mathbf{C}$ is defined as $\mathrm{d}^{(1)}(\mathbf{x}, \mathbf{C}) = \min_{\mathbf{c}\in\mathbf{C}}(1 - \langle \mathbf{x}, \mathbf{c}\rangle)$. *Fisher-mean* of a set of vectors $\mathcal{X} = \{\mathbf{x}\}$ is denoted as $\mathrm{FM}(\mathcal{X})$:

$$\mathrm{FM}(\mathcal{X}) = \frac{\sum_{\mathbf{x}\in\mathcal{X}} \mathbf{x}}{||\sum_{\mathbf{x}\in\mathcal{X}} \mathbf{x}||}.$$

2.1. Spherical k-means clustering

Let $\mathcal{X} = \{\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_n\}$ denote a set of n vectors on the unit sphere in $\mathbb{R}^d$, that is $||\mathbf{x}_i|| = 1$ for $1 \leq i \leq n$. Let $\{\pi_1, \pi_2, \ldots, \pi_k\}$ denote a partition of vectors into k disjoint clusters such that

$$\bigcup_{i=1}^{k} \pi_i = \mathcal{X} \text{ and } \pi_i \cap \pi_j = \phi \text{ for } i \neq j.$$

For each $1 \leq i \leq k$ the Fisher-mean of the vectors belong to the cluster π_i is $\mathbf{c}_i = \frac{\sum_{\mathbf{x}\in\pi_i} \mathbf{x}}{||\sum_{\mathbf{x}\in\pi_i} \mathbf{x}||}$. The quality of any partitioning $\bigcup_{i=1}^{k}\{\pi_i\}$ is evaluated using the following objective function

$$\Phi_{\mathbf{C}}^{(1)}(\mathcal{X}) = \sum_{i=1}^{k} \sum_{\mathbf{x}\in\pi_i} \mathrm{d}^{(1)}(\mathbf{x}, \mathbf{c}_i) = |\mathcal{X}| - \sum_{i=1}^{k} \sum_{\mathbf{x}\in\pi_i} \langle \mathbf{x}, \mathbf{c}_i \rangle. \tag{1}$$

Spherical k-means involves finding a partitioning of the given set $\mathcal{X}$ so that the above objective function is minimized. Clearly, minimizing $\Phi_{\mathbf{C}}^{(1)}(\mathcal{X})$ is equivalent to maximization of $\sum_{i=1}^{k} \sum_{\mathbf{x}\in\pi_i} \langle \mathbf{x}, \mathbf{c}_i \rangle$. This problem has been shown to be NP-Complete (Kleinberg et al., 1998). We denote $\mathbf{c}_i^*$ as the optimal cluster center of i-th cluster and $\mathrm{OPT} = \{\mathbf{c}_i^*\}_{i=1}^{k}$ as the set of k optimal cluster centers.

An algorithm for this problem was proposed by (Dhillon and Modha, 2001). We call it SPKM. There are two steps in the algorithm – the first step consist of uniformly sampling k vectors – initial cluster centers – and is called as *seeding step*; in the second step (*Llyods-type iterations*) each vector is assigned to a cluster center having smallest angular distance, and then new cluster centers are computed by calculating Fisher-mean of the vectors belonging to that cluster. The second step is repeated until a stopping criterion is satisfied – when there is little to no improvement in the clustering cost.

2.2. Extension of dissimilarity measure in Spherical k-means

For two unit vectors $\mathbf{x}$ and $\mathbf{c}$, the *extended angular dissimilarity* between them is defined as $\mathrm{d}^{(\alpha)}(\mathbf{x}, \mathbf{c}) = \alpha - \langle \mathbf{x}, \mathbf{c} \rangle$, where $\alpha \geq 3/2$. Similarly, we define the *α–Spherical k-means* clustering objective function as follows:

$$\Phi_{\mathbf{C}}^{(\alpha)}(\mathcal{X}) = \sum_{i=1}^{k} \sum_{\mathbf{x}\in\pi_i} \mathrm{d}^{(\alpha)}(\mathbf{x}, \mathbf{c}_i) = \alpha|\mathcal{X}| - \sum_{i=1}^{k} \sum_{\mathbf{x}\in\pi_i} \langle \mathbf{x}, \mathbf{c}_i \rangle. \tag{2}$$

Minimization of $\Phi_{\mathbf{C}}^{(\alpha)}(\mathcal{X})$ is equivalent to maximization of $\sum_{i=1}^{k} \sum_{\mathbf{x}\in\pi_i} \langle \mathbf{x}, \mathbf{c}_i \rangle$. Thus, both the objective functions mentioned in Equations 1,2 describe the same clustering problem. In a first look this extension in the dissimilarity measure looks meaningless (Endo and Miyamoto, 2015), though it carries a nice property – dissimilarity measure satisfies the triangle inequality – which is crucial for proving correctness of *Spherical k-means++* algorithm. We state it in the lemma below:

Lemma 2 (Lemma 1 of (Endo and Miyamoto, 2015)) *$\mathrm{d}^{(\alpha)}(,)$ satisfies the triangle inequality, when $\alpha \geq 3/2$.*

2.3. Spherical k-mean++ (SPKM++) (Endo and Miyamoto, 2015)

A drawback of the algorithm proposed by (Dhillon and Modha, 2001) is that it may converge to a local minima. (Endo and Miyamoto, 2015) proposed an adaptive sampling algorithm which samples one data point in each iteration, and in total samples k points (seeding step) such that the clustering obtained by considering them as cluster centers gives an $O(\log k)$ competitive result, with respect to the optimal clustering. Llyods-type iteration step further improves the clustering quality. We discuss their sampling strategy as follows: the first vector is sampled uniformly at random from the given set of vectors. The remaining $k-1$ points are sequentially added to previously sampled centers based on the following probability distribution

$$p(\mathbf{x}|\mathbf{C}) = \frac{\mathrm{d}^{(\alpha)}(\mathbf{x},\mathbf{C})}{\sum_{\mathbf{x}'\in\mathcal{X}} \mathrm{d}^{(\alpha)}(\mathbf{x}',\mathbf{C})} = \frac{\mathrm{d}^{(\alpha)}(\mathbf{x},\mathbf{C})}{\Phi_{\mathbf{C}}^{(\alpha)}(\mathcal{X})}, \tag{3}$$

where $\mathrm{d}^{(\alpha)}(\mathbf{x},\mathbf{C}) = \min_{\mathbf{c}\in\mathbf{C}}(\alpha - \langle \mathbf{x},\mathbf{c}\rangle)$, where $\alpha \geq 3/2$. We call the above sampling approach as *angular-sampling*. We present a theoretical guarantee on their sampling algorithm as follows:

Theorem 4 (Theorem 2 of (Endo and Miyamoto, 2015)) *Let $\mathcal{X}$ be a set of n vectors in a d-dimensional unit sphere, $\epsilon \in (0,1)$, and $\mathbf{C}$ be a set consisting of initially sampled k-points, where the first point is sampled uniformly at random and the remaining $k-1$ points are sampled via angular-sampling as stated in Equation 3. Then*

$$\mathbb{E}[\Phi_{\mathbf{C}}^{(\alpha)}(\mathcal{X})] \leq 4(\log k + 2)\Phi_{\mathrm{OPT}}(\mathcal{X}).$$

Note 3 *For simplicity in notation, we drop α from the notation of dissimilarity measure and clustering objective, and consider the value $\alpha = 3/2$ throughout.*

3. Our results (SPKM-MCMC) – sampling k initial seeding points *via* Markov chain

3.1. Approximating angular-sampling:

As mentioned earlier, the SPKM++ algorithm suggests that sampling initial data points following the angular-sampling distribution (see, Equation 3) results in a clustering cost which is within an $O(\log k)$ factor from the optimal clustering result. However, sampling from such a distribution requires taking k passes over data, which might be sub-optimal when the value of k, and/or n, d are large. Our goal is to reduce the complexity by approximating angular-sampling, i.e, we wish to obtain a faster sampling algorithm whose implied sampling probabilities $\bar{p}(x)$ are close to the underlying angular sampling distribution $p(x)$. In order to measure the closeness of the distributions, we use the total-variation distance, which is defined below.

Definition 5 (Total Variation) *Let Ω be a finite sample space on which two probability distributions p and q are defined. The total variation distance $||p-q||_{TV}$ between p and q is defined as $\frac{1}{2}\sum_{x\in\Omega}|p(x)-q(x)|$.*

In what follows, using total variation distance, we give a bound on the solution quality of our proposed algorithm. Roughly speaking, if the total variation distance between our sampling distribution and underlying angular-sampling distribution is less than ϵ, then a similar clustering guarantee as in (Endo and Miyamoto, 2015) is maintained, with probability of at least $1 - \epsilon$.

Algorithm 1: Markov chain based initial seeding for Spherical k-means clustering.

Input: Data set $\mathcal{X}$, chain-length m, number of clusters k.
Output: A set of initial cluster centers (seeding points) $\mathbf{C} = \{\mathbf{c_1}, \mathbf{c_2}, \ldots, \mathbf{c_k}\}$.

Preprocessing step:
$\mathbf{c_1} \leftarrow$ a vector sampled uniformly at random from $\mathcal{X}$.
for $x \in \mathcal{X}$ **do**
  $q(\mathbf{x}|\mathbf{c_1}) = \frac{\mathrm{d}(\mathbf{x},\mathbf{c_1})}{2\sum_{\mathbf{x}'\in\mathcal{X}} \mathrm{d}(\mathbf{x}',\mathbf{c_1})} + \frac{1}{2|\mathcal{X}|}$
end
Main algorithm:
$\mathbf{C} \leftarrow \{\mathbf{c_1}\}$
for $i = 2, 3, \ldots, k$ **do**
  $x \leftarrow$ point sampled from $q(x)$
  $d_x \leftarrow \mathrm{d}(x, \mathbf{C})$
  for $j = 2, 3, \ldots, m$ **do**
    $y \leftarrow$ point sampled from $q(y)$
    $d_y \leftarrow \mathrm{d}(y, \mathbf{C})$
    if $\frac{d_y q(x)}{d_x q(y)} > \mathrm{Unif}(0, 1)$ **then**
      $x \leftarrow y, d_x \leftarrow d_y$
    end
  end
  $\mathbf{C} \leftarrow \mathbf{C} \cup \{x\}$
end

3.2. Approximating angular-sampling via Markov chain:

We propose an alternate way of sampling initial k concept vectors *via* a Markov chain, which closely approximates the underlying angular sampling distribution, and requires only one pass of the data. We first uniformly sample one vector $\mathbf{c_1}$ from the set of vectors $\mathcal{X}$ and then iteratively build a Markov chain. In each iteration $j \in [2, \ldots, m]$, where m is the chain length, we uniformly sample a candidate $y_j \sim q(x)$ with the following probability.

$$q(\mathbf{x}|\mathbf{c_1}) = \frac{\mathrm{d}(\mathbf{x}, \mathbf{c_1})}{2\sum_{\mathbf{x}'\in\mathcal{X}} \mathrm{d}(\mathbf{x}', \mathbf{c_1})} + \frac{1}{2|\mathcal{X}|} = \frac{\mathrm{d}(\mathbf{x}, \mathbf{c_1})}{2\ \Phi_{\mathbf{c_1}}(\mathcal{X})} + \frac{1}{2|\mathcal{X}|}, \tag{4}$$

Then, we either accept $x_j \leftarrow y_j$ with probability

$$\pi(x_{j-1}, y_j) = \min\left(\frac{p(y_j)}{p(x_{j-1})}\frac{q(x_{j-1})}{q(y_j)}, 1\right),$$

or reject it $x_j \leftarrow x_{j-1}$, where $p(.|\mathbf{C})$ is the probability as mentioned in the Equation 3. For a Markov chain of length m, we need to calculate the distance between m data points and their respetive cluster centers. The stationary distribution of this Markov chain is the target distribution $p(x)$, that is, $\bar{p}_m(x)$ of the m-th state x_m converges to $p(x)$. Corollary 1 of (Cai, 2000) quantifies it and shows that the total variation distance decreases at a geometric rate with respect to the chain length m.

$$||p(.|\mathbf{C}) - \bar{p_m}(.|\mathbf{C})||_{\text{TV}} \leq \left(1 - \frac{1}{\gamma}\right)^{m-1}$$

where $\gamma = \max_{\mathbf{x} \in \mathcal{X}} \frac{p(\mathbf{x}|\mathbf{C})}{q(\mathbf{x}|\mathbf{c_1})}$. This suggest that chain length $m = O(\gamma \log \frac{1}{\epsilon})$ achieves a total variation of at most ϵ.

We summarize our algorithm in Algorithm 1 and its correctness follows from Theorem 1 [2] .

Intution: In our proposed algorithm, we sample the first point $\mathbf{c_1}$ uniformly from $\mathcal{X}$, and based on $\mathbf{c_1}$, we recall our proposal distribution below:

$$q(\mathbf{x}|\mathbf{c_1}) = \frac{\mathrm{d}(\mathbf{x}, \mathbf{c_1})}{2 \sum_{\mathbf{x}' \in \mathcal{X}} \mathrm{d}(\mathbf{x}', \mathbf{c_1})} + \frac{1}{2|\mathcal{X}|} = \frac{\mathrm{d}(\mathbf{x}, \mathbf{c_1})}{2\ \Phi_{\mathbf{c_1}}(\mathcal{X})} + \frac{1}{2|\mathcal{X}|}$$

The proposed distribution has two parts – the first part is based on the *angular-distribution* (as stated in Equation 3) with respect to $\mathbf{c_1}$, which is the best possible distribution for the second iteration. We show that this distribution also suffices for later iterations. The second term works as a regularization term which ensures that the mixing time of the Markov chain is always within a factor of two.

There are three key steps in our analysis – we first bound how well a single Markov chain approximates one iteration of exact angular sampling, then show that even for later iterations the Markov chain distribution is close to the underlying angular-sampling distribution under total variation distance (see Definition 5). Finally, a proof shows how the approximation error accumulates across iterations, and give a bound on the expected solution quality. We start with the following lemma which shows that in any iteration $\Phi_{\mathbf{C}}(\mathcal{X})$ is ϵ_1 competitive with respect to $\Phi_{\mathbf{c_1}}(\mathcal{X})$, or how well a Markov chain distribution approximates angular sampling distribution under total variation distance. In the following $\bar{p_m}(\mathbf{x}|\mathbf{C})$ denotes probability of sampling a point $\mathbf{x} \in \mathcal{X}$ *via* a Markov chain of length m.

Lemma 6 *Let $\mathbf{C} \subseteq \mathcal{X}$, with $\mathbf{c_1} \in \mathbf{C}$, where $\mathbf{c_1}$ is the first point sampled in Algorithm 1 line 2. For $\epsilon_1, \epsilon_2 \in (0, 1)$, $m \geq 1 + \frac{2}{\epsilon_1} \log \frac{1}{\epsilon_2}$, then at least one of the following condition holds:*

1. $\Phi_{\mathbf{C}}(\mathcal{X}) < \epsilon_1 \Phi_{\mathbf{c_1}}(\mathcal{X})$,
2. $||p(.|\mathbf{C}) - \bar{p_m}(.|\mathbf{C})||_{\text{TV}} \leq \epsilon_2$.

2. Our proposed algorithm and its analysis are similar to (Bachem et al., 2016) which proposed a Markov chain based initial seeding for k-means. They generalize their result to other clustering problems and to any metric space for which there exists a global isometry on Euclidean space. In this work, we show that a similar analysis also works for Spherical k-means clustering problem, where the underlying dissimilarity measure – *"1- cosine similarity"* – does not satisfy the triangle inequality, which is a key requirement for a dissimilarity measure to be a metric space.

Proof Consider a fixed $\mathbf{c_1}$ and $\mathbf{C}$ with $\mathbf{c_1} \in \mathbf{C}$. If Condition 1 holds, then we are done. We give proof of a lemma assuming that Condition 1 doesn't hold, i.e, $\Phi_{\mathbf{C}}(\mathcal{X}) \geq \epsilon_1 \Phi_{\mathbf{c_1}}(\mathcal{X})$. By its design the stationarity distribution of the Markov chain mentioned in Algorithm 1 is $p(.|\mathbf{C})$. Due to Corollary 1 of (Cai, 2000), total variation distance between the two distributions is bounded by

$$||p(.|\mathbf{C}) - p_m^-(.|\mathbf{C})||_{\text{TV}} \leq \left(1 - \frac{1}{\gamma}\right)^{m-1} \leq e^{-\frac{m-1}{\gamma}} \tag{5}$$

where $\gamma = \max_{\mathbf{x} \in \mathcal{X}} \frac{p(\mathbf{x}|\mathbf{C})}{q(\mathbf{x}|\mathbf{c_1})}$. If chain length $m \geq 1 + \gamma \log \frac{1}{\epsilon_2}$, then due to Equation 5, the total variation distance is bounded by ϵ_2. We give a bound on γ now due to Equations 3,4:

$$\begin{aligned}\gamma = \max_{\mathbf{x} \in \mathcal{X}} \frac{p(\mathbf{x}|\mathbf{C})}{q(\mathbf{x}|\mathbf{c_1})} &\leq \frac{\mathrm{d}(\mathbf{x}, \mathbf{C})}{\Phi_{\mathbf{C}}(\mathcal{X})} \Big/ \left(\frac{\mathrm{d}(\mathbf{x}, \mathbf{c_1})}{2\ \Phi_{\mathbf{c_1}}(\mathcal{X})} + \frac{1}{2|\mathcal{X}|}\right) \\ &\leq \frac{\mathrm{d}(\mathbf{x}, \mathbf{C})}{\Phi_{\mathbf{C}}(\mathcal{X})} \Big/ \frac{\mathrm{d}(\mathbf{x}, \mathbf{c_1})}{2\ \Phi_{\mathbf{c_1}}(\mathcal{X})} = 2 \frac{\mathrm{d}(\mathbf{x}, \mathbf{C})}{\mathrm{d}(\mathbf{x}, \mathbf{c_1})} . \frac{\Phi_{\mathbf{c_1}}(\mathcal{X})}{\Phi_{\mathbf{C}}(\mathcal{X})} \leq \frac{2}{\epsilon_1}\end{aligned}$$

The final inequality follows due to our assumption $\Phi_{\mathbf{C}}(\mathcal{X}) \geq \epsilon_1 \Phi_{\mathbf{c_1}}(\mathcal{X})$ and the fact that $\mathrm{d}(\mathbf{x}, \mathbf{C}) \leq \mathrm{d}(\mathbf{x}, \mathbf{c_1})$. ■

We now give a bound on the expected clustering cost by adding points through Markov chain sampling. Let $\mathbf{c_1}$ be the vector sampled by the Algorithm 1, and let $\mathbf{C}$ be a set of already sampled vectors with $\mathbf{c_1} \in \mathbf{C}$. Let us denote $A^{\mathbf{c_1}}(\mathbf{C}, l)$ as the expected clustering cost after sequentially adding $l \in \mathbb{N}$ points to already sampled set of centers $\mathbf{C}$ *via* Markov chain as stated in Algorithm 1 in lines 11–17. Thus, by definition we have,

$$A^{\mathbf{c_1}}(\mathbf{C}, l) = \sum_{\mathbf{x} \in \mathcal{X}} p_m^-(\mathbf{x}|\mathbf{C}) A^{\mathbf{c_1}}(\mathbf{C} \cup \{\mathbf{x}\}, l-1),$$

where $A^{\mathbf{c_1}}(\mathbf{C}, 0) = \Phi_{\mathbf{C}}(\mathcal{X})$. Define $P^{\mathbf{c_1}}(\mathbf{C}, l)$ as the probability of sampling a solution which is ϵ_1-competitive with respect to $\Phi_{\mathbf{c_1}}(\mathcal{X})$ after adding l vectors to $\mathbf{C}$ *via* Markov chain as stated in Algorithm 1.

$$\begin{aligned}P^{\mathbf{c_1}}(\mathbf{C}, l) &= \mathbf{1}_{\Phi_{\mathbf{C}}(\mathcal{X}) < \epsilon_1 \Phi_{\mathbf{c_1}}(\mathcal{X})}, \\ P^{\mathbf{c_1}}(\mathbf{C}, l) &= \sum_{\mathbf{x} \in \mathcal{X}} p_m^-(\mathbf{x}|\mathbf{C}) P^{\mathbf{c_1}}(\mathbf{C} \cup \{\mathbf{x}\}, l-1).\end{aligned}$$

We finally define $B(\mathbf{C}, l)$ as expected clustering cost of sequentially adding l vectors to an already sampled set of vectors $\mathbf{C}$ *via* angular-sampling.

$$B(\mathbf{C}, l) = \sum_{\mathbf{x} \in \mathcal{X}} p(\mathbf{x}|\mathbf{C}) B(\mathbf{C} \cup \{\mathbf{x}\}, l-1),$$

where $B(\mathbf{C}, 0) = A^{\mathbf{c_1}}(\mathbf{C}, 0) = \Phi_{\mathbf{C}}(\mathcal{X}), \forall \mathbf{c_1} \in \mathcal{X}$.

The following lemma relates $B(\mathbf{C}, l)$ with $A^{\mathbf{c_1}}(\mathbf{C}, l)$ which helps to relates the clustering cost of adding points *via* angular-sampling *vs.* Markov chain sampling. A proof of the lemma below follows from induction over l and Lemma 6. Proof arguments of the lemma below are in the similar lines as of Lemma 2 of (Bachem et al., 2016), we defer it to the full version of the paper.

Lemma 7 *Let* $\mathbf{C} \subseteq \mathcal{X}$*, with* $\mathbf{c_1} \in \mathbf{C}$*, where* $\mathbf{c_1}$ *is the first point sampled in Algorithm* 1 *line* 2*. For* $\epsilon_1, \epsilon_2 \in (0,1), m \geq 1 + \frac{2}{\epsilon_1} \log \frac{1}{\epsilon_2}, l \in \mathbb{N}$*, the following holds:*

$$A^{\mathbf{c_1}}(\mathbf{C}, l) \leq (\epsilon_1 P^{\mathbf{c_1}}(\mathbf{C}, l) + l\epsilon_2)\Phi_{\mathbf{c_1}}(\mathcal{X}) + B(\mathbf{C}, l)$$

Putting it all together – Proof of Theorem 1:

Using Lemma 6 and 7 we completes a proof of Theorem 1. Let $\mathbf{C} = \{\mathbf{c_1}\}$, and $l = k - 1$, then due to Lemma 7, we have the following:

$$\begin{aligned} A^{\mathbf{c_1}}(\mathbf{C}, l) &\leq (\epsilon_1 P^{\mathbf{c_1}}(\mathbf{C}, k-1) + (k-1)\epsilon_2)\Phi_{\mathbf{c_1}}(\mathcal{X}) + B(\mathbf{C}, k-1) \\ &\leq (\epsilon_1 + (k-1)\epsilon_2)\,\Phi_{\mathbf{c_1}}(\mathcal{X}) + B(\mathbf{C}, k-1) \qquad (6) \\ &= \frac{\epsilon}{2}\Phi_{\mathbf{c_1}}(\mathcal{X}) + B(\mathbf{C}, k-1) \qquad (7) \end{aligned}$$

Inequality 6 holds as $P^{\mathbf{c_1}}(\mathbf{C}, k-1) \leq 1$; and Equality 7 holds by choosing $\epsilon_1 = \epsilon/2$ and $\epsilon_2 = \epsilon/4k$. By Lemma 4 of (Endo and Miyamoto, 2015) we have,

$$\frac{1}{|\mathcal{X}|} \sum_{\mathbf{c_1} \in \mathcal{X}} \Phi_{\mathbf{c_1}}(\mathcal{X}) \leq 2\mathrm{AV}(\mathcal{X}), \qquad (8)$$

and Theorem 1 of (Endo and Miyamoto, 2015) we have,

$$\frac{1}{|\mathcal{X}|} \sum_{\mathbf{c_1} \in \mathcal{X}} B(\mathbf{C}, k-1) \leq 4(\log k + 2)\Phi_{\mathrm{OPT}}(\mathcal{X}) \qquad (9)$$

Equation 7 along with Equations 8, 9 completes a proof of the Theorem.

Complexity of the algorithm: the pre-processing step (between line number $1-5$) of Algorithm 1 requires taking one pass, and has $O(nd)$ complexity. For each iteration $i = \{2, 3, ..., k\}$, the complexity of constructing the Markov chain is $O(im)$, leading to complexity of the main loop $O(\frac{1}{\epsilon}k^2 d \log \frac{k}{\epsilon})$.

4. Applications of the results

4.1. Text clustering

Document clustering has become ubiquitous and is a fundamental problem in identifying latent factors, automatic topic extraction, filtering, fast information retrieval etc. Clustering is an unsupervised learning problem where the goal is to determine the intrinsic grouping in a set of unlabeled data. As *cosine similarity* is a widely accepted choice for computing similarity between two text documents (represented in vectors using any possible representations such as BoW (Bag-of-words), *tf-idf* etc.), the Spherical k-means clustering algorithm proposed by Dhillon and Modha is a more appropriate formulation for clustering text documents (Dhillon and Modha, 2001). Cosine similarity can be efficiently computed for sparse vectors, the SPKM algorithm exploits the sparsity of text data and quickly converges to a local minima. As our proposed clustering algorithm gives a faster and closer to optimal clustering for Spherical k-means clustering problem, it naturally becomes a better choice than SPKM for clustering text documents.

4.2. Scaling-up Recommendation systems

Collaborative filtering algorithms are widely used among existing approaches to recommender systems. Traditional approaches include assigning a user to the cluster containing the most similar users, and then using the purchases and ratings of users in this cluster to generate recommendations. Model based techniques like clustering are very often used to represent users and items by means of a d-dimensional latent factor space (Su and Khoshgoftaar, 2009). Here, the similarity is measured by "Pearson Correlation Coefficient" or by "Cosine Similarity" (Vozalis and Margaritis, 2003). Since clustering forms an important part in the collaborative filtering algorithm, there has been sufficient work attempting to achieve this in a spherical setting. Previous work shows that spherical k-means clustering on the Netflix dataset resulted in a more meaningful clustering of movies into genres (Ampazis, 2008). This method has also shown better performance in terms of both accuracy and computational costs. Other recommender systems with better performance also make use of SPKM for generation of clustering ensembles (Castro-Herrera et al., 2009). Work aimed at identifying tags in recommender systems has made extensive use of the SPKM algorithm (Hayes and Avesani, 2007). Recommender systems in changing environments like evolving online forums also make use of SPKM for clustering (Castro-Herrera et al., 2009). As our proposed clustering algorithm gives a faster and closer to optimal clustering for Spherical k-means, in conjunction with the approaches mentioned above, it will lead to an efficient, accurate, and scalable algorithm for recommendation systems.

4.3. Nearest neighbour search

Our proposed algorithm can be used to efficiently solve an approximate nearest neighbor search for inner product. Given a set of data points (vectors) and a query vector, in the K-MIPS problem, the goal is to pick the top K vectors that maximises their dot product with the query vector. (Auvolat et al., 2015) proposed an approach to solve the K-MIPS problem via Spherical k-means clustering. It consist of first reducing the problem to K-MCSS (maximum cosine similarity search) by padding both input and query vectors using the asymmetric LSH algorithm of (Shrivastava and Li, 2014) and then perform Spherical k-means clustering on top of them. To find the top K vectors that have maximum cosine similarity with the query point, they first find the cluster whose centroid has the highest cosine similarity with the query vector, then they consider all the points belonging to that cluster as a candidate set. If there are n points, then clustering them into $\sqrt{n}$ clusters leads each cluster to contain approximately $\sqrt{n}$ points. Thus, the search time drops roughly from $O(n)$ to $O(\sqrt{n})$. As our proposed clustering algorithm gives a faster and closer to optimal clustering for Spherical k-means, in conjunction with the approach of (Auvolat et al., 2015), will lead to an efficient and accurate algorithm for the K-MIPS problem.

Along with the above mentioned applications, spherical k-means clustering is a central subroutine in many other fundamental machine learning applications such as dimension reduction (Dhillon and Modha, 2001), learning feature representation (Coates and Ng, 2012), hypertext clustering and web searching (Modha and Spangler, 2000), spam filtering (Delany et al., 2012), document summarization (Dunlavy et al., 2007), non-negative matrix factorization (Wild, 2002). Our proposed algorithm can potentially lead to faster and accurate algorithms in such applications.

Table 1: Real-world dataset description.

Data Set	No. of documents	No. of words in the vocab (dimension)	Max no. of words in a document(sparsity)
NIPS full papers	1500	12419	914
KOS blog entries	3430	6906	457
BBC	9635	2225	128
20 Newsgroups	1700	56916	734

5. Experiments

Hardware description: All experiments were run on a standard Intel - Xeon 3.5GHz Quad-Core Processor and Corsair - Vengeance 2 x 16GB Memory. [3]

Datasets: The experiments were performed on publicly available datasets - NIPS full papers (Lichman, 2013), KOS blog entries (Lichman, 2013), BBC (Greene and Cunningham, 2006) and 20Newsgroups (Lang). After tokenization and removal of stopwords, the vocabulary of unique words was truncated by only keeping words that occurred more than ten times. These datasets are "BoW"(Bag-of-words) representations of the corresponding text corpora. For those words that do not occur in the document, their frequency were taken to be zero. Their cardinality, dimension, and sparsity is in Table 1.

5.1. Experimental comparison between SPKM and SPKM++

Experimental setup: We first pre-process the datasets and normalise each document vector such that it has a unit norm. In order to evaluate the clustering cost, we use the cost function mentioned in Equation 2. Let us recall the equation below:

$$\Phi_{\mathbf{C}}(\mathcal{X}) = \sum_{i=1}^{k} \sum_{\mathbf{x} \in \pi_i} \mathrm{d}(\mathbf{x}, \mathbf{c}_i) = \alpha |\mathcal{X}| - \sum_{i=1}^{k} \sum_{\mathbf{x} \in \pi_i} \langle \mathbf{x}, \mathbf{c}_i \rangle,$$

where $\mathcal{X}$ is the set of documents, and $\alpha = 3/2$. In this setting a lower clustering cost implies better clustering quality and hence is better. SPKM and SPKM++ were run for different values of $k \in \{3, 5, 7, 10\}$. For SPKM, k initial seeding points were randomly sampled, and iterated over using Lloyd's algorithm. The iterations are terminated when there is less than 0.001 improvement in cost from the previous iteration. For SPKM++ we only perform the seeding step – sample k points sequentially *via* the angular sampling distribution (see Equation 2). Clustering cost is calculated immediately after the seeding step – assuming those k-sampled vectors as concept vectors and assigning all the points to the one that has the highest cosine similarity. In order to reduce the effect of randomness, we repeat each experiment 5 times and consider the average.

Insight: The experiments were performed on all datasets and results for KOS and BBC datasets are in Tables 2, 3 respectively. Similar results were obtained on the other two datasets but we do not include them due to space constraints. As visible from Tables 2, 3, it is clear that SPKM++ gives better/comparable clustering results right after seeding when compared to SPKM obtained at the end of Lloyd's iterations. For example: on KOS dataset,

3. Implementations of SPKM, SPKM++, and SPKM-MCMC are available at the following repository: https://github.com/Prat-123/SPKM.

Table 2: Comparison of clustering quality between SPKM and SPKM++ for different values of k on KOS dataset.

k	SPKM (Clustering cost)	SPKM++ (Seeding cost)	SPKM (Total running time(s))	SPKM++) (Seeding time(s))
3	3974.64	3958.73	159.17	9.64
5	3832.52	3832.57	369.67	31.65
7	3788.65	3781.33	531.34	64.66
10	3747.02	3736.83	616.64	139.93

Table 3: Comparison of clustering quality between SPKM and SPKM++ for different values of k on BBC dataset.

k	SPKM (Clustering cost)	SPKM++ (Seeding cost)	SPKM (Total running time(s))	SPKM++) (Seeding time(s))
3	13344.63	13350.20	286.95	8.63
5	13209.44	13184.78	526.33	28.38
7	13068.64	13080.07	1056.70	59.17
10	12925.35	12941.94	1350.85	125.89

when $k = 3$, SPKM++ obtained 16 times speed-up in the running time with respect to SPKM, while simultaneously achieving improvement in the clustering quality. Of course, clustering results of SPKM++ will further improve if Lloyd's iterations are applied after the results of seeding step.

5.2. Experimental comparison between SPKM++ and SPKM-MCMC

Experimental setup: We empirically validate our theoretical results and compare our proposed algorithm Spherical k-Means MCMC (SPKM-MCMC) with that of SPKM++. For the SPKM-MCMC algorithm, different chain lengths $m \in \{5, 30, 100, 500\}$ were considered. Both the algorithms, SPKM++ and SPKM-MCMC, were run on all datasets and clustering quality and seeding time were recorded.

Insight: Tables 4 and 5 provide a comparison of seeding time and clustering quality by considering the speed-up and clustering quality improvement offered by SPKM-MCMC relative to SPKM++. On all datasets, the seeding time for SPKM-MCMC is significantly lower than that of SPKM++, for all values of m. The seeding time increases with m, but

Table 4: Comparison of seeding time between SPKM++ and SPKM-MCMC for different values of m. Entries of the table describe the relative speed-up in the seeding time with respect to the seeding time of SPKM++.

$k = 10$	KOS	BBC	NIPS	20News
SPKM++	1	1	1	1
SPKM-MCMC (m=5)	×8.0	×7.5	×5.4	×4.8
SPKM-MCMC (m=30)	×7.6	×7.0	×5.0	×3.3
SPKM-MCMC (m=100)	×6.6	×5.7	×4.2	×1.8
SPKM-MCMC (m=500)	×4.0	×2.7	×2.2	×0.5

Table 5: Comparison of clustering cost between SPKM++ and SPKM-MCMC for different values of m. Entries of the table describe the relative improvement the clustering cost with respect to the cost of SPKM++.

$k = 10$	KOS	BBC	NIPS	20News
SPKM++	0.00%	0.00%	0.00%	0.00%
SPKM-MCMC (m=5)	-0.03%	0.07%	0.08%	0.48%
SPKM-MCMC (m=30)	-0.07%	-0.03%	0.08%	0.03%
SPKM-MCMC (m=100)	-0.06%	-0.03%	0.09%	-0.14%
SPKM-MCMC (m=500)	-0.43%	0.06%	-0.13%	-0.08%

Table 6: Comparison of seeding time between SPKM++ and SPKM-MCMC for different values of m. Entries of the table describe the relative speed-up in the seeding time with respect to the seeding time of SPKM++.

$k = 30$	KOS	BBC	NIPS	20News
SPKM++	1	1	1	1
SPKM-MCMC (m=5)	×26.4	×26.4	× 25.9	× 19.8
SPKM-MCMC (m=30)	×23.5	×23.5	× 21.1	× 8.5
SPKM-MCMC (m=100)	×18.0	×17.2	× 13.8	× 3.3
SPKM-MCMC (m=500)	×7.6	×6.8	× 4.6	× 0.7

Table 7: Comparison of clustering cost between SPKM++ and SPKM-MCMC for different values of m. Entries of the table describe the relative improvement the clustering cost with respect to the cost of SPKM++.

$k = 30$	KOS	BBC	NIPS	20News
SPKM++	0.00%	0.00%	0.00%	0.00%
SPKM-MCMC (m=5)	0.01%	-0.01 %	-0.01 %	0.97 %
SPKM-MCMC (m=30)	-0.06%	-0.02 %	-0.11 %	1.16 %
SPKM-MCMC (m=100)	0.19%	0.08 %	-0.08%	1.4 %
SPKM-MCMC (m=500)	-0.32%	0.35 %	-0.31 %	0.66 %

Table 8: Comparison of seeding time between SPKM++ and SPKM-MCMC for different values of m. Entries of the table describe the relative speed-up in the seeding time with respect to the seeding time of SPKM++.

$k = 50$	KOS	BBC	NIPS	20News
SPKM++	1	1	1	1
SPKM-MCMC (m=5)	×45.22	×43.93	× 43.34	× 39.6
SPKM-MCMC (m=30)	×37.82	×37.42	× 31	× 22.7
SPKM-MCMC (m=100)	×25.91	×25.19	× 17.34	× 13.1
SPKM-MCMC (m=500)	×9.4	×8.97	× 4.9	× 5.2

Table 9: Comparison of clustering cost between SPKM++ and SPKM-MCMC for different values of m. Entries of the table describe the relative improvement the clustering cost with respect to the cost of SPKM++.

$k = 50$	KOS	BBC	NIPS	20News
SPKM++	0.00%	0.00%	0.00%	0.00%
SPKM-MCMC (m=5)	-0.16%	2.81 %	-0.5 %	-0.68 %
SPKM-MCMC (m=30)	-0.1%	2.34 %	-0.46 %	0.34 %
SPKM-MCMC (m=100)	0.05%	2.62 %	0.16%	0.55 %
SPKM-MCMC (m=500)	-0.2%	2.80 %	-0.42 %	1.2 %

is compensated with the clustering quality improving and quickly converging to the the clustering quality produced by SPKM++. Even with a small chain length, SPKM-MCMC produces clusters that are close to that of SPKM++ at a fraction of the computational cost of seeding. For example: on the KOS dataset, for $k = 10$ and $m = 5$, SPKM-MCMC obtains an $8\times$ speed-up (seeding time) with respect to that of SPKM++, while simultaneously maintaining the clustering quality within 0.03% of SPKM++.

We run the experiments for two different values of $k \in \{10, 30, 50\}$, and obtain a higher speed-up with respect to SPKM++ for higher values of k. As mentioned earlier, the reason being that SPKM++ requires taking k passes over data, while SPKM-MCMC takes only one pass. This observation is reflected in our experiments. For example: on the KOS dataset, when the value of chain length $m = 5$, we obtained an $8\times$ speed-up for $k = 10$, a $26.4\times$ speed-up for $k = 30$ and a $45.22\times$ speed-up for $k = 50$. Moreover, clustering costs produced by SPKM-MCMC are also approximately preserved in both the scenarios. This observation makes our proposed algorithm (SPKM-MCMC) a more appropriate choice when the values of k are large.

6. Concluding Remarks

We experimentally validate the SPKM++ (Endo and Miyamoto, 2015) on publicly available datasets and show that it outperforms the state-of-the-art algorithm SPKM (Dhillon and Modha, 2001) for the Spherical k-means clustering problem. We proposed a Markov chain based sampling algorithm for initial seeding of k data points. We obtained significant speed up in the seeding time as our sampling algorithm requires taking only one pass over the data as compared to k passes required by SPKM++. In terms of the clustering cost, we retained an $O(\log k)$ multiplicative approximation guarantee with respect to the optimal clustering result, similar to SPKM++. Our algorithm only includes an extra additive term which depends on the *angular variance* of the dataset. We experimentally evaluated our algorithm on public datasets and obtained a significant speed-up with respect to seeding time of SPKM++ while maintaining almost the same clustering quality. The speed-up in the seeding time is more prominent as the value of k increases. Our proposed algorithm is simple and easy to implement. Therefore it can easily be adapted in practice.

References

Nicholas Ampazis. Collaborative filtering via concept decomposition on the netflix dataset. In *ECAI*, volume 8, pages 26–30, 2008.

David Arthur and Sergei Vassilvitskii. k-means++: The advantages of careful seeding. In *Proceedings of the eighteenth annual ACM-SIAM symposium on Discrete algorithms*, pages 1027–1035. Society for Industrial and Applied Mathematics, 2007.

Alex Auvolat, Sarath Chandar, Pascal Vincent, Hugo Larochelle, and Yoshua Bengio. Clustering is efficient for approximate maximum inner product search. *arXiv preprint arXiv:1507.05910*, 2015.

Olivier Bachem, Mario Lucic, Seyed Hamed Hassani, and Andreas Krause. Fast and provably good seedings for k-means. In *Advances in Neural Information Processing Systems 29: Annual Conference on Neural Information Processing Systems 2016, December 5-10, 2016, Barcelona, Spain*, pages 55–63, 2016. URL http://papers.nips.cc/paper/6478-fast-and-provably-good-seedings-for-k-means.

Haiyan Cai. Exact bound for the convergence of metropolis chains. *Stochastic Analysis and Applications*, 18(1):63–71, 2000.

Carlos Castro-Herrera, Chuan Duan, Jane Cleland-Huang, and Bamshad Mobasher. A recommender system for requirements elicitation in large-scale software projects. In *Proceedings of the 2009 ACM symposium on Applied Computing*, pages 1419–1426. ACM, 2009.

Adam Coates and Andrew Y. Ng. Learning feature representations with k-means. In *Neural Networks: Tricks of the Trade - Second Edition*, pages 561–580. 2012. doi: 10.1007/978-3-642-35289-8_30. URL https://doi.org/10.1007/978-3-642-35289-8_30.

Sarah Jane Delany, Mark Buckley, and Derek Greene. Sms spam filtering. *Expert Syst. Appl.*, 39(10):9899–9908, August 2012. ISSN 0957-4174. doi: 10.1016/j.eswa.2012.02.053. URL http://dx.doi.org/10.1016/j.eswa.2012.02.053.

Inderjit S. Dhillon and Dharmendra S. Modha. Concept decompositions for large sparse text data using clustering. *Machine Learning*, 42(1/2):143–175, 2001. doi: 10.1023/A:1007612920971. URL https://doi.org/10.1023/A:1007612920971.

Daniel M. Dunlavy, Dianne P. O'Leary, John M. Conroy, and Judith D. Schlesinger. QCS: A system for querying, clustering and summarizing documents. *Inf. Process. Manage.*, 43(6):1588–1605, 2007. doi: 10.1016/j.ipm.2007.01.003. URL https://doi.org/10.1016/j.ipm.2007.01.003.

Yasunori Endo and Sadaaki Miyamoto. Spherical k-means++ clustering. In *Modeling Decisions for Artificial Intelligence - 12th International Conference, MDAI 2015, Skövde, Sweden, September 21-23, 2015, Proceedings*, pages 103–114, 2015. doi: 10.1007/978-3-319-23240-9_9. URL https://doi.org/10.1007/978-3-319-23240-9_9.

Derek Greene and Pádraig Cunningham. Practical solutions to the problem of diagonal dominance in kernel document clustering. In *Proc. 23rd International Conference on Machine learning (ICML'06)*, pages 377–384. ACM Press, 2006.

Conor Hayes and Paolo Avesani. Using tags and clustering to identify topic-relevant blogs. In *Proceedings of the First International Conference on Weblogs and Social Media, ICWSM 2007, Boulder, Colorado, USA, March 26-28, 2007*, 2007. URL http://www.icwsm.org/papers/paper23.html.

Mark Hill, Colin A Harrower, Christopher D Preston, et al. Spherical k-means clustering is good for interpreting multivariate species occurrence data. *Methods in Ecology and Evolution*, 4(6):542–551, 2013.

Jon M. Kleinberg, Christos H. Papadimitriou, and Prabhakar Raghavan. A microeconomic view of data mining. *Data Min. Knowl. Discov.*, 2(4):311–324, 1998. doi: 10.1023/A:1009726428407. URL https://doi.org/10.1023/A:1009726428407.

K. Lang. 20 newsgroups data set. URL http://www.ai.mit.edu/people/jrennie/20Newsgroups/.

M. Lichman. UCI machine learning repository, 2013. URL http://archive.ics.uci.edu/ml.

S. Lloyd. Least squares quantization in pcm. volume 28, pages 129–137, Piscataway, NJ, USA, September 2006. IEEE Press. doi: 10.1109/TIT.1982.1056489. URL http://dx.doi.org/10.1109/TIT.1982.1056489.

Dharmendra S. Modha and W. Scott Spangler. Clustering hypertext with applications to web searching. In *Proceedings of the Eleventh ACM on Hypertext and Hypermedia*, HYPERTEXT '00, pages 143–152, New York, NY, USA, 2000. ACM. ISBN 1-58113-227-1. doi: 10.1145/336296.336351. URL http://doi.acm.org/10.1145/336296.336351.

Anshumali Shrivastava and Ping Li. Asymmetric LSH (ALSH) for sublinear time maximum inner product search (MIPS). In *Advances in Neural Information Processing Systems 27: Annual Conference on Neural Information Processing Systems 2014, December 8-13 2014, Montreal, Quebec, Canada*, pages 2321–2329, 2014. URL http://papers.nips.cc/paper/5329-asymmetric-lsh-alsh-for-sublinear-time-maximum-inner-product-search-mips.

Xiaoyuan Su and Taghi M. Khoshgoftaar. A survey of collaborative filtering techniques. *Adv. Artificial Intellegence*, 2009:421425:1–421425:19, 2009. doi: 10.1155/2009/421425. URL https://doi.org/10.1155/2009/421425.

Emmanouil Vozalis and Konstantinos G Margaritis. Analysis of recommender systems algorithms. In *The 6th Hellenic European Conference on Computer Mathematics & its Applications*, pages 732–745, 2003.

Stefan Wild. Seeding non-negative matrix factorizations with spherical k-means clustering. *Masters thesis, University of Colorado*, 2002.