Semi-Supervised Algorithms for Approximately Optimal and Accurate Clustering

Buddhima Gamlath
École Polytechnique Fédérale de Lausanne, Lausanne, Switzerland
buddhima.gamlath@epfl.ch

Sangxia Huang
Sony Mobile Communications, Lund, Sweden
huang.sangxia@gmail.com

Ola Svensson
École Polytechnique Fédérale de Lausanne, Lausanne, Switzerland
ola.svensson@epfl.ch

Abstract

We study k-means clustering in a semi-supervised setting. Given an oracle that returns whether two given points belong to the same cluster in a fixed optimal clustering, we investigate the following question: how many oracle queries are sufficient to efficiently recover a clustering that, with probability at least $(1-\delta)$, simultaneously has a cost of at most $(1+\epsilon)$ times the optimal cost and an accuracy of at least $(1-\epsilon)$?

We show how to achieve such a clustering on n points with $O((k^2 \log n) \cdot m(Q, \epsilon^4, \delta/(k \log n)))$ oracle queries, when the k clusters can be learned with an ϵ' error and a failure probability δ' using $m(Q, \epsilon', \delta')$ labeled samples in the supervised setting, where Q is the set of candidate cluster centers. We show that $m(Q, \epsilon', \delta')$ is small both for k-means instances in Euclidean space and for those in finite metric spaces. We further show that, for the Euclidean k-means instances, we can avoid the dependency on n in the query complexity at the expense of an increased dependency on k: specifically, we give a slightly more involved algorithm that uses $O(k^4/(\epsilon^2\delta) + (k^9/\epsilon^4)\log(1/\delta) + k \cdot m(\mathbb{R}^r, \epsilon^4/k, \delta))$ oracle queries.

We also show that the number of queries needed for $(1-\epsilon)$-accuracy in Euclidean k-means must linearly depend on the dimension of the underlying Euclidean space, and for finite metric space k-means, we show that it must at least be logarithmic in the number of candidate centers. This shows that our query complexities capture the right dependencies on the respective parameters.

2012 ACM Subject Classification Theory of computation → Facility location and clustering

Keywords and phrases Clustering, Semi-supervised Learning, Approximation Algorithms, k-Means, k-Median

Digital Object Identifier 10.4230/LIPIcs.ICALP.2018.57

Related Version The full version of this work is available at `https://arxiv.org/abs/1803.00926`.

Funding This research was supported by ERC Starting Grant 335288-OptApprox.

45th International Colloquium on Automata, Languages, and Programming (ICALP 2018).
Editors: Ioannis Chatzigiannakis, Christos Kaklamanis, Dániel Marx, and Donald Sannella;
Article No. 57; pp. 57:1–57:14

Leibniz International Proceedings in Informatics
Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

1 Introduction

Clustering is a fundamental problem that arises in many learning tasks. Given a set P of data points, the goal is to output a k-partition $C_1 \dot\cup \ldots \dot\cup C_k$ of P according to some optimization criteria. In unsupervised clustering, the data points are unlabeled. The classic k-means problem and other well-studied clustering problems such as k-median fall into this category.

In a general k-means clustering problem, the input comprises a finite set of n points P that is to be clustered, a set of candidate centers Q, and a distance metric d giving the distances between each pair of points in $P \cup Q$. The goal is to find k cluster centers $c_1, \ldots, c_k \in Q$ that minimizes the cost, which is the sum of squared distances between each point in P and its closest cluster center. In this case, the clustering $\mathcal{C}$ is defined by setting $C_i = \{x \in P : c_i \text{ is the closest center to } x\}$ for all $i = 1, \ldots, k$ and breaking ties arbitrarily. Two widely studied special cases are the k-means problem in Euclidean space (where $P \subset \mathbb{R}^r, Q = \mathbb{R}^r$, and d is the Euclidean distance function) and the k-means problem in finite metric spaces (where $(P \cup Q, \mathrm{d})$ forms a finite metric space).

Despite its popularity and success in many settings, there are two known drawbacks of the unsupervised k-means problem:

1. Finding the centers that satisfy the clustering goal is computationally hard. For example, even the special case of 2-means problem in Euclidean space is NP-hard [8].
2. There could be multiple possible sets of centers that minimize the cost. However, in practical instances, not all such sets are equally meaningful, and we would like our algorithm to find one that corresponds to the concerns of the application.

Since k-means is NP-hard, it is natural to seek approximation algorithms. For the general k-means problem in Euclidean space, notable approximation results include the local search by Kanungo et al. [13] with an approximation guarantee of $(9 + \epsilon)$ and the recent LP-based 6.357-approximation algorithm by Ahmadian et al. [1]. On the negative side, Lee et al. [14] ruled out arbitrarily good approximation algorithms for the k-means problem on general instances. For several special cases, however, there exist PTASes. For example, in the case where k is constant, Har-Peled and Mazumdar [11] and Feldman et al. [9] showed how to get a PTAS using weak coresets, and in the case where the dimension d is constant, Cohen-Addad et al. [7] and Friggstad et al. [10] gave PTASes based on a basic local search algorithm. In addition, Awasthi et al. [4] presented a PTAS for k-means, assuming that the input is "clusterable" (satisfies a certain stability criterion).

Even if we leave aside the computational issues with unsupervised k-means, we still have the problem that there can be multiple different clusterings that minimize the cost. To see this, consider the 2-means problem on the set of vertices of an equilateral triangle. In this case, we have three different clusterings that give the same minimum cost, but only one of the clusterings might be meaningful. One way to avoid this issue is to have strong assumptions on the input. For example, Balcan et al. [5] considered the problem in a restricted setting where any c-approximation to the problem also classifies at least a $(1 - \epsilon)$ fraction of the points correctly.

Ashtiani et al. [3] recently proposed a different approach for addressing the aforementioned drawbacks. They introduced a semi-supervised active clustering framework where the algorithm is allowed to make queries of the form *same-cluster(x, y)* to a domain expert, and the expert replies whether the points x and y belong to the same cluster in some fixed optimal clustering. Under the additional assumptions that the clusters are contained inside k balls in $\mathbb{R}^r$ that are sufficiently far away from each other, they presented an algorithm that makes $O(k^2 \log k + k(\log n + \log(1/\delta)))$ same-cluster queries, runs in $O(kn \log n + k^2 \log(1/\delta))$ time,

and recovers the clusters with probability at least $(1-\delta)$. Their algorithm finds approximate cluster centers, orders all points by their distances to the cluster centers, and performs binary searches to determine the radii of the balls. Although it recovers the exact clusters, this approach works only when the clusters are contained inside well-separated balls. When the clusters are determined by a general Voronoi partitioning, and thus distances to the cluster boundaries can differ in different directions, this approach fails.

A natural question arising from the work of Ashtiani et al. [3] is whether such strong assumptions on the input structure are necessary. Ailon et al. [2] addressed this concern and considered the problem without any assumptions on the structure of the underlying true clusters. Their main result was a polynomial-time $(1+\epsilon)$-approximation scheme for k-means in the same semi-supervised framework as in Ashtiani et al. [3]. However, in contrast to Ashtiani et al. [3], their work gives no assurance on the accuracy of the recovered clustering compared to the true clustering. To achieve their goal, the authors utilized importance sampling to uniformly sample points from small clusters that significantly contribute to the cost. Their algorithm makes $O(k^9/\epsilon^4)$ same-cluster queries, runs in $O(nr(k^9/\epsilon^4))$ time, and succeeds with a constant probability.

In this work, we investigate the k-means problem in the same semi-supervised setting as Ailon et al. [2], but in addition to approximating the cost, we seek a solution that is also accurate with respect to the true clustering. We assume that the underlying true clustering minimizes the cost, and that there are no points on cluster boundaries (i.e., the margin between each pair of clusters can be arbitrarily small but not zero). This last assumption is what differentiates our setup from that of Ailon et al. [2]. It is reasonable to assume that no point lies on the boundary of two clusters, as otherwise, to achieve constant accuracy, we would have to query at least a constant fraction of the boundary points. Without querying each boundary point, we have no way of determining to which cluster it belongs.

Observe that if we label all the points correctly with respect to the true clustering, the resulting clustering automatically achieves the optimal cost. However, such perfect accuracy is difficult to achieve as there may be points that are arbitrarily close to each other but belong to different clusters. Using only a reasonable number of samples, the best we can hope for is to recover an approximately accurate solution. PAC (Probably Approximately Correct) learning helps us achieve this goal and provides a trade-off between the desired accuracy and the required number of samples.

Suppose that we have a solution where only a small fraction of the input points is incorrectly classified. In this case, one would hope that the cost is also close to the optimal cost. Unfortunately, the extra cost incurred by the incorrectly classified points can be very high depending on their positions, true labels, and the labels assigned to them. Our main concern in this paper is controlling this additional cost.

We show that if we start with a constant-factor approximation for the cost, we can refine the clustering using a PAC learning algorithm. This yields a simple polynomial-time algorithm that, given a k-means instance and $(\epsilon, \delta) \in (0,1)^2$ as parameters, with probability at least $(1-\delta)$ outputs a clustering that has a cost of at most $(1+\epsilon)$ times the optimal cost and that classifies at least a $(1-\epsilon)$ fraction of the points correctly with respect to the underlying true clustering. To do so, the algorithm makes $O((k^2 \log n) \cdot m(Q, \epsilon^4, \delta/(k \log n)))$ same-cluster queries. Here, $m(Q, \epsilon', \delta')$ is the sufficient number of labeled samples for a PAC learning algorithm to learn k clusters in a k-means instance with an ϵ' error and a failure probability δ' in the supervised setting (recall that Q is the set of candidate centers). We further show that our algorithm can be easily adapted to k-median and other similar problems that use the ℓ'th power of distances in place of squared distances for some fixed

$\ell > 0$. We formally present this result as Theorem 6 in Section 3. In Theorem 1 below, we give an informal statement for the case of k-means.

▶ **Theorem 1** (An informal version of Theorem 6). *There exists a semi-supervised learning algorithm that, given a k-means instance, oracle access to same-cluster queries that are consistent with some fixed optimal clustering, and parameters $(\epsilon, \delta) \in (0,1)^2$, outputs a clustering that, with probability at least $(1-\delta)$, correctly labels (up to a permutation of the labels) at least a $(1-\epsilon)$ fraction of the points and, simultaneously, has a cost of at most $(1+\epsilon)$ times the optimal cost. In doing so, the algorithm makes $O((k^2 \log n) \cdot m(Q, \epsilon^4, \delta/(k \log n)))$ same-cluster queries.*

Our algorithm is general and applicable to any family of k-means, k-median, or similar distance based clustering instances that can be efficiently learned with PAC learning. As discussed later in this work, these include Euclidean and general finite metric space clustering instances. In contrast, both Ashtiani et al. [3] and Ailon et al. [2], considered only the Euclidean k-means problem. To the best of our knowledge, ours is the first such result applicable to finite metric space k-means and both Euclidean and finite metric space k-median problems.

Ideally, we want $m(Q, \epsilon, \delta)$ to be small. Additionally, the analysis of our algorithm relies on two natural properties of learning algorithms. Firstly, we require PAC learning to always correctly label all the sampled points. Secondly, we also require it to not 'invent' new labels and only output labels that it has seen on the samples. We show that such learning algorithms with small $m(Q, \epsilon, \delta)$ exist both for k-means instances in Euclidean space and for those in finite metric spaces with no points on the boundaries of the optimal clusters. For r-dimensional Euclidean k-means, $m(Q = \mathbb{R}^r, \epsilon, \delta)$ has a linear dependency on r. For the case of finite metric spaces, $m(Q, \epsilon, \delta)$ has a logarithmic dependency on $|Q|$, which is the size of the set of candidate centers. In fact, these learning algorithms are applicable not only to k-means instances but also to instances of other similar center-based clustering problems (where clusters are defined by assigning points to their closest cluster centers). We discuss our learning algorithms in detail in the full version of this paper.

Our semi-supervised learning algorithm is inspired by the work of Feldman et al. [9] on weak coresets. Their construction of the weak coresets first obtains an intermediate clustering using a constant-factor approximation algorithm and refines each intermediate cluster by taking random samples. In order to get a good guarantee for the cost, their algorithm partitions each cluster into an inner ball that contains the majority of the points, and an outer region that contains the remaining points. We proceed similarly to this construction; however, we further partition the outer region into $O(\log n)$ concentric rings and use PAC learning to label the points in the inner ball and in each of the outer rings separately. For Euclidean k-means instances, the number of same-cluster queries needed by the algorithm has a logarithmic dependency on the number n of points, which is similar (up to a $\text{poly}(\log \log n)$ factor) to that of the algorithm by Ashtiani et al. [3]. The advantage of our algorithm is that it works for a much broader range of k-means instances whereas the applicability of the algorithm of Ashtiani et al. [3] is restricted to those instances whose clusters are contained in well-separated balls in Euclidean space.

This algorithm is effective in many natural scenarios where the number of clusters k is larger than $\log n$. However, as the size of the k-means instance (i.e., the number of points) becomes large, the $\log n$ factor becomes undesirable. In Euclidean k-means, the number of samples needed by the learning algorithm for an ϵ error and a failure probability δ does not depend on n. The $\log n$ dependency in the final query complexity is exclusively due to repeating the PAC learning step on $\Omega(k \log n)$ different partitions of P. To overcome

this problem, we present a second algorithm, which is applicable only to Euclidean k-means instances, inspired by the work of Ailon et al. [2]. This time, we start with a $(1+\epsilon)$-approximation for the cost and refine it using PAC learning. Unlike our first algorithm, we only run the PAC learning once on the whole input, and thus we completely eliminate the dependency on n. The disadvantages of this algorithm compared to our first algorithm are the slightly more involved nature of the algorithm and the increased dependency on k in its query complexity. Theorem 2 below formally states this result. We present our algorithm in Section 4 and discuss the key ideas that lead to its construction. The complete proof of Theorem 2 is given in the full version.

▶ **Theorem 2.** *There exists a polynomial-time algorithm that, given a k-means instance in r-dimensional Euclidean space, oracle access to same-cluster queries that are consistent with some fixed optimal clustering, and parameters $(\epsilon, \delta) \in (0,1)^2$, outputs a clustering that, with probability at least $(1-\delta)$, correctly labels (up to a permutation of the labels) at least a $(1-\epsilon)$ fraction of the points and, simultaneously, has a cost of at most $(1+\epsilon)$ times the optimal cost. The algorithm makes $O(k^4/(\epsilon^2\delta) + (k^9/\epsilon^4)\log(1/\delta) + k \cdot m(\mathbb{R}^r, \epsilon^4/k, \delta))$ same-cluster queries.*

For the Euclidean setting, the query complexities of both our algorithms have a linear dependency on the dimension of the Euclidean space. The algorithm of Ashtiani et al. [3] does not have such a dependency due to their strong assumption on the cluster structure, whereas the one by Ailon et al. [2] does not have that as it only approximates the cost. We show that, in our scenario, such a dependency is necessary to achieve the accuracy guarantees of our algorithms. For the finite metric space k-means, the query complexity of our general algorithm has an $O(\text{poly}(\log|P|, \log|Q|))$ dependency. The dependency on $|P|$ comes from the repeated application of the learning algorithm on $\Omega(k \log|P|)$ different partitions, and whether we can avoid this is an open problem. However, we show that an $\Omega(\log|Q|)$ query complexity is necessary for the accuracy. We formalize these results in Theorem 3 below (the proof is in the full version).

▶ **Theorem 3.** *Let K be a family of k-means instances. Let $\mathcal{A}$ be an algorithm that, given a k-means instance in K, oracle access to same-cluster queries for some fixed optimal clustering, and parameters $(\epsilon, \delta) \in (0,1)^2$, outputs a clustering that, with probability at least $(1-\delta)$, correctly labels (up to a permutation of the cluster labels) at least a $(1-\epsilon)$ fraction of the points. Then, the following statements hold:*

1. *If K is the family of k-means instances in r-dimensional Euclidean space that have no points on the boundaries of optimal clusters, $\mathcal{A}$ must make $\Omega(r)$ same-cluster queries.*
2. *If K is the family of finite metric space k-means instances that have no points on the boundaries of optimal clusters, $\mathcal{A}$ must make $\Omega(\log|Q|)$ same-cluster queries.*

The outline of this extended abstract is as follows. In Section 2 we introduce the notation, formulate the problem and present the learning theorems that we use in the subsequent sections. In Section 3 we present our first algorithm, which is simple and applicable to general k-means instances that admit efficient learning algorithms, but has a dependency of $\log n$ in its query complexity. Finally, in Section 4 we discuss how to remove the $\log n$ dependency in the query complexity for the special case of Euclidean k-means instances and present our second algorithm.

In the full version, we present formal proofs of all the stated results, where we also introduce the basic concepts and tools of PAC learning and explain how to design learning algorithms for Euclidean and finite metric space k-means instances.

2 Preliminaries

In this section, we introduce the basic notation and two common families of k-means instances, and formally define the k-means problem that we address in this work. We also introduce the notion of *learnability* for families of k-means instances and state two learning theorems that will be used in the later sections.

2.1 k-Means Problem in a Semi-supervised Setting

Let P and Q be two sets of points where $|P| = n$, and let $\mathrm{d} : (P \cup Q) \times (P \cup Q) \to \mathbb{R}_+$ be a distance metric. We denote a k-means instance by the triple (P, Q, d). Two common families of k-means instances we consider in this work are:

1. k-means instances in Euclidean space, where $P \subset \mathbb{R}^r$, $Q = \mathbb{R}^r$, and $\mathrm{d}(x_1, x_2) = \|x_1 - x_2\|$ is the Euclidean distance between x_1 and x_2, and
2. k-means instances in finite metric spaces, where $(P \cup Q, \mathrm{d})$ forms a finite metric space.

Let $[k] := \{1, \ldots, k\}$. We identify a k-clustering $\mathcal{C}$ of (P, Q, d) by a labeling function $f_{\mathcal{C}} : P \to [k]$, and a set of k centers, $c_1, \ldots, c_k \in Q$, associated with each label, $1, \ldots, k$. For each label $i \in [k]$ of a clustering $\mathcal{C}$, let $C_i := \{p \in P : f_{\mathcal{C}}(p) = i\}$ be the set of points whose label is i. For convenience, we may use the labeling function $f_{\mathcal{C}}$ or the set of clusters $\{C_1, \ldots, C_k\}$ interchangeably to denote a clustering $\mathcal{C}$.

For a subset $C \subseteq P$ and a point $q \in Q$, define $\mathrm{cost}(C, q) := \sum_{p \in C} \mathrm{d}^2(p, q)$. For each i, define center $c_i := \mathrm{argmin}_{q \in Q} \mathrm{cost}(C_i, q)$, i.e., each center is a point in Q that minimizes the sum of squared distances between itself and each of the points assigned to it. For a k-clustering $\mathcal{C}$, we define its *k-means cost* as $\mathrm{cost}(\mathcal{C}) := \sum_{i \in [k]} \mathrm{cost}(C_i, c_i)$. Let $\mathcal{C}^*$ be the set of all k-clusterings of (P, Q, d). Then, the optimal k-means cost of (P, Q, d) is defined as $OPT := \min_{\mathcal{C} \in \mathcal{C}^*} \mathrm{cost}(\mathcal{C})$. We say that a k-clustering $\mathcal{C}$ α-approximates the k-means cost if $\mathrm{cost}(\mathcal{C}) \leq \alpha OPT$.

Let $\mathcal{O}$ be a fixed k-clustering of (P, Q, d) that achieves the optimal k-means cost, and let $\mathcal{C}$ be any k-clustering of P. Let $f_{\mathcal{O}}$ and $f_{\mathcal{C}}$ be the labeling functions that correspond to $\mathcal{O}$ and $\mathcal{C}$ respectively. We assume that we have oracle access to the labeling function $f_{\mathcal{O}}$ of the optimal target clustering up to a permutation of the labels. We can simulate a single query to such an oracle with $O(k)$ queries to a same-cluster oracle (see the full version for details). A same-cluster oracle is an oracle that answers *same-cluster*(p_1, p_2) queries with 'yes' or 'no' based on whether p_1 and p_2 belong to the same cluster in the fixed optimal clustering $\mathcal{O}$.

The error of a clustering $\mathcal{C}$ with respect to the clustering $\mathcal{O}$ for a k-means instance (P, Q, d) is now defined as $\mathrm{error}(\mathcal{C}, \mathcal{O}) := \min_{\sigma} |\{p \in P : f_{\mathcal{O}}(p) \neq \sigma(f_{\mathcal{C}}(p))\}|$, where the minimization is over all permutations $\sigma : [k] \to [k]$. In other words, $\mathrm{error}(\mathcal{C}, \mathcal{O})$ is the minimum number of points incorrectly labeled by the clustering $\mathcal{C}$ with respect to the optimal clustering $\mathcal{O}$, considering all possible permutations of the cluster labels. The reason for defining error in this manner is because we use a simulated version of $f_{\mathcal{O}}$ (which is only accurate up to a permutation of the cluster labels) instead of the true $f_{\mathcal{O}}$ to learn cluster labels. We say that a k-clustering $\mathcal{C}$ is $(1 - \alpha)$-accurate with respect to $\mathcal{O}$ if $\mathrm{error}(\mathcal{C}, \mathcal{O}) \leq \alpha n$.

Given (P, Q, d), parameters k and $(\epsilon, \delta) \in (0, 1)^2$, and oracle access to $f_{\mathcal{O}}$, our goal is to output a k-clustering $\hat{\mathcal{O}}$ of (P, Q, d) that, with probability at least $(1 - \delta)$, satisfies $\mathrm{error}(\hat{\mathcal{O}}, \mathcal{O}) \leq \epsilon n$ and $\mathrm{cost}(\hat{\mathcal{O}}) \leq (1 + \epsilon) OPT$.

2.2 PAC Learning for k-Means

Let K be a family of k-means instances, and let $m(Q, \epsilon, \delta)$ be a positive integer-valued function. We say such a family K is *learnable* with *sample complexity* m if there exists a learning algorithm $\mathcal{A}_L$ such that the following holds: Let $\epsilon \in (0,1)$ be an error parameter and let $\delta \in (0,1)$ be a probability parameter. Let (P, Q, d) be a k-means instance that belongs to K. Let $\mathcal{O}$ be a fixed optimal k-means clustering and let $f_{\mathcal{O}}$ be the associated labeling function. Let T be a fixed subset of P, and let S be a multiset of at least $m(Q, \epsilon, \delta)$ independently and uniformly distributed samples from T. The algorithm $\mathcal{A}_L$, given input (P, Q, d) and $(s, f_{\mathcal{O}}(s))$ for all $s \in S$, outputs a function $h : P \to [k]$. Moreover, with probability at least $(1-\delta)$ over the choice of S, the output h agrees with $f_{\mathcal{O}}$ on at least a $(1-\epsilon)$ fraction of the points in T (i.e., $|\{p \in T : h(p) = f_{\mathcal{O}}(p)\}| \geq (1-\epsilon)|T|$). This simpler notion of learnability is sufficient for the purpose of this work although it deviates from that of the general PAC learnability, which concerns with samples drawn from arbitrary distributions.

We say that such a learning algorithm $\mathcal{A}_L$ has the *zero sample error* property if the output h of $\mathcal{A}_L$ assigns the correct label to all the sampled points (i.e., $h(s) = f_{\mathcal{O}}(s)$ for all $s \in S$). Furthermore, we say that such a learning algorithm $\mathcal{A}_L$ is *non-inventive* if it does not 'invent' labels that it has not seen. This means that the output h of $\mathcal{A}_L$ does not assign labels that were not present in the input (sample, label) pairs (i.e., if $h(x) = c$ for some $x \in P$, then for some sample point $s \in S$, $f_{\mathcal{O}}(s) = c$).

In Section 3, we present a simple algorithm for $(1+\epsilon)$-approximate and $(1-\epsilon)$-accurate k-means clustering for a family K of k-means instances, assuming that K is learnable with a zero sample error, non-inventive learning algorithm. In the analysis, zero sample error and non-inventive properties play a key role in the crucial step of bounding the cost of incorrectly labeled points in terms of that of correctly labeled nearby points.

We now present two learning theorems for the Euclidean setting and the finite metric space setting (see the full version for the proofs). Assuming no point lies on cluster boundaries, the theorems state that the labeling function $f_{\mathcal{O}}$ of the optimal clustering is learnable with a zero sample error, non-inventive learning algorithm in both settings. We say that a k-means instance (P, Q, d) has *no boundary points* if in any optimal clustering $\mathcal{O}$ with clusters $O_1, \ldots, O_k$ and respective centers $o_1, \ldots, o_k$, the closest center to any given point $p \in P$ is unique (i.e., if $p \in O_i$, $\mathrm{d}(p, o_i) < \mathrm{d}(p, o_j)$ for all $j \neq i$).

▶ **Theorem 4** (Learning k-Means in Euclidean Space). *Let* $\mathrm{d}(p_1, p_2) = \|p_1 - p_2\|$ *be the Euclidean distance function. Let* $K = \{(P, \mathbb{R}^r, \mathrm{d}) : P \subset \mathbb{R}, |P| < \infty, (P, \mathbb{R}^r, \mathrm{d})$ *has no boundary points*$\}$ *be the family of* k*-means instances that are in* r*-dimensional Euclidean space and that have no boundary points. The family* K *is learnable with sample-complexity*[1] $m(\mathbb{R}^r, \epsilon, \delta) = \tilde{O}((k^2 r \log(k^2 r)(\log(k^3 r/\epsilon)) + \log(1/\delta))/\epsilon)$.

▶ **Theorem 5** (Learning k-Means in Finite Metric Spaces). *Let* $K = \{(P, Q, \mathrm{d}) : (P \cup Q, \mathrm{d})$ *is a finite metric space, and* (P, Q, d) *has no boundary points*$\}$ *be the family of finite metric space* k*-means instances that have no boundary points. The family* K *is learnable with sample-complexity*[2] $m(Q, \epsilon, \delta) = \tilde{O}((k^2(\log k)(\log |Q|)(\log k + \log 1/\epsilon) + \log(1/\delta))/\epsilon)$.

[1] $\tilde{O}$ hides $\mathrm{poly}(\log\log k, \log\log r)$ factors.
[2] $\tilde{O}$ hides $\mathrm{poly}(\log\log k, \log\log |Q|)$ factors.

3 A Simple Algorithm for $(1+\epsilon)$ Cost and $(1-\epsilon)$ Accuracy

Let K be a family of k-means instances that is learnable with sample complexity m using a zero sample error, non-inventive learning algorithm $\mathcal{A}_L$. Let $\mathcal{A}_\alpha$ be a constant-factor approximation algorithm (in terms of cost) for k-means, and let $\mathcal{A}_1$ be a polynomial-time algorithm for the 1-means problem (i.e., given $(P, Q, \mathrm{d}) \in K$, $\mathcal{A}_1$ finds $\operatorname{argmin}_{q \in Q} \operatorname{cost}(P, q)$ in polynomial time). We present a simple semi-supervised learning algorithm that, given a k-means instance (P, Q, d) of class K and oracle access to the labeling function $f_{\mathcal{O}}$ of a fixed optimal clustering $\mathcal{O}$ of (P, Q, d), outputs a clustering $\hat{\mathcal{O}}$ that, with probability at least $(1-\delta)$, satisfies $\operatorname{cost}(\hat{\mathcal{O}}) \leq (1+\epsilon)OPT$ and $\operatorname{error}(\hat{\mathcal{O}}, \mathcal{O}) \leq \epsilon|P|$. Our algorithm uses $\mathcal{A}_\alpha$, $\mathcal{A}_1$, and $\mathcal{A}_L$ as subroutines and makes $O((k \log |P|) \cdot m(Q, \epsilon^4, \delta/(k \log |P|)))$ oracle queries. We show that our algorithm can be easily modified for $(1+\epsilon)$-approximate and $(1-\epsilon)$-accurate k-median and other similar distance-based clustering problems. Towards the end of this section, we discuss several applications of this result, namely, for Euclidean and finite metric space k-means and k-median problems.

Let us start by applying the learning algorithm $\mathcal{A}_L$ to learn all the cluster labels. If we get perfect accuracy, the cost will be optimal. A natural question to ask in this case is: what happens to the cost if the learning output has ϵ error? In general, even a single misclassified point can incur an arbitrarily large additional cost. To better understand this, consider the following: Let $O_i, O_j \subseteq P$ be two distinct optimal clusters in the target clustering, and let o_i, o_j be their respective cluster centers. Let $p \in O_i$ be a point that is incorrectly classified and assigned label $j \neq i$ by $\mathcal{A}_L$. Also assume that the number of misclassified points is small enough so that the centers of the clusters output by the learning algorithm are close to those of the optimal clustering. Thus, in the optimal clustering, p incurs a cost of $\mathrm{d}^2(p, o_i)$, whereas according to the learning outcome, p incurs a cost that is close to $\mathrm{d}^2(p, o_j)$. In the worst case, $\mathrm{d}(p, o_j)$ can be arbitrarily larger than $\mathrm{d}(p, o_i)$.

Now suppose that, within distance ρ from p, there exists some point $q \in O_j$. In this case, we can bound the cost incurred due to the erroneous label of p using the true cost of p in the target clustering. To be more specific, using the triangle inequality, we get the following bound for any metric space: $\mathrm{d}(p, o_j) \leq \mathrm{d}(p, q) + \mathrm{d}(q, o_j) \leq \rho + \mathrm{d}(q, o_j)$. Furthermore, due to the optimality, $\mathrm{d}(q, o_j) \leq \mathrm{d}(q, o_i) \leq \mathrm{d}(q, p) + \mathrm{d}(p, o_i) \leq \rho + \mathrm{d}(p, o_i)$. Hence, it follows that $\mathrm{d}(p, o_j) \leq 2\rho + \mathrm{d}(p, o_i)$. To utilize this observation in an algorithmic setting, we need to make sure that, for every point that is misclassified into cluster j, there exists a correctly classified nearby point q that belongs to the optimal cluster O_j. Luckily, this is ensured by the combination of zero sample error and non-inventive properties of $\mathcal{A}_L$. If a point is misclassified into cluster j, the non-inventive property says that $\mathcal{A}_L$ must have seen a sample point q from cluster j. The zero sample error property ensures that q is labeled correctly by $\mathcal{A}_L$. To make sure that such correctly labeled points are sufficiently close to their incorrectly labeled counterparts, we run $\mathcal{A}_L$ separately on certain suitably bounded partitions of P.

The formal description of our algorithm is given in Algorithm 1. The outline is as follows: First, we run $\mathcal{A}_\alpha$ on (P, Q, d) and obtain an intermediate clustering $\mathcal{C} = \{C_1, \ldots, C_k\}$. For each C_i, we run $\mathcal{A}_1$ to find a suitable center c_i. Next, we partition each intermediate cluster C_i into an inner ball and $O(\log |P|)$ outer rings centered around c_i. We run the learning algorithm $\mathcal{A}_L$ separately on each of these partitions. We choose the inner and outer radii of the rings so that, in each partition, the points that are incorrectly classified by the learning algorithm only incur a small additional cost compared to that of the correctly classified points. The final output is a clustering $\hat{\mathcal{O}}$ that is consistent with the learning outputs on each of the partitions. For each cluster $\hat{O}_i$, we associate the output of running $\mathcal{A}_1$ on $(\hat{O}_i, Q, \mathrm{d})$ as

Algorithm 1: A simple algorithm for $(1+\epsilon)$-approximate $(1-\epsilon)$-accurate k-means.

Input : k-Means instance (P, Q, d), oracle access to $f_{\mathcal{O}}$, constant-factor approximation algorithm $\mathcal{A}_\alpha$ for k-means, 1-means algorithm $\mathcal{A}_1$, learning algorithm $\mathcal{A}_L$ with sample complexity m, accuracy parameter $0 < \epsilon < 1$, and failure probability $0 < \delta < 1$.

Output : The clustering $\hat{\mathcal{O}} = \{\hat{O}_1, \ldots, \hat{O}_k\}$ defined by the labeling $f_{\hat{\mathcal{O}}} : P \to [k]$ computed below. The respective cluster centers are $\hat{o}_i = \operatorname{argmin}_{q \in Q} \operatorname{cost}(\hat{O}_i, q)$, which can be found by running $\mathcal{A}_1$ on $(\hat{O}_i, Q, \mathrm{d})$.

1 Let $n = |P|$, and let $\gamma = \epsilon^2/(288\alpha)$.
2 Run $\mathcal{A}_\alpha$ and obtain an α-approximate k-means clustering $\mathcal{C} = \{C_1, \ldots, C_k\}$. For each $i \in [k]$, run $\mathcal{A}_1$ on (C_i, Q, d) and find centers $c_i = \operatorname{argmin}_{q \in Q} \operatorname{cost}(C_i, q)$.
3 **for** $C_i \in \mathcal{C}$ **do**
4 Let $r_i = \sqrt{\operatorname{cost}(C_i, c_i)/(\gamma |C_i|)}$.
5 Let $C_{i,0}$ be all points in C_i that are at most r_i away from c_i.
6 Let $C_{i,j}$ be the points in C_i that are between $2^{j-1} r_i$ and $2^j r_i$ away from c_i for $j = 1, \ldots, (\log n)/2$.
7 Let $m' = m(Q, \gamma^2, \delta/(k \log n))$.
8 **for** *each non-empty* $C_{i,j}$ **do**
9 Sample m' points $x_1, \ldots, x_{m'} \in C_{i,j}$ independently and uniformly at random.
10 Query the oracle on $x_1, \ldots, x_{m'}$ and let $S_{i,j} = \{(x_i, f_{\mathcal{O}}(x_i)) : i = 1, \ldots, m'\}$.
11 Run $\mathcal{A}_L$ on input (P, Q, d) and $S_{i,j}$, and obtain a labeling $h_{i,j} : C_{i,j} \to [k]$.
12 Output the clustering $\hat{\mathcal{O}}$ defined by the following labeling function:
13 **for** *each* i, j, $x \in C_{i,j}$ **do**
14 Set $f_{\hat{\mathcal{O}}}(x) = h_{i,j}(x)$.

its center. Note that, due to the accuracy requirements, the cluster center to which a point is assigned in the output may not be the cluster center closest to that point in the output. It remains an interesting problem to find an accurate clustering in which every point is always assigned to its nearest cluster center.

With probability at least $(1-\delta)$, Algorithm 1 outputs a $(1+\epsilon)$-approximately optimal, $(1-\epsilon)$-accurate k-means clustering (the complete analysis is in the full version). In Algorithm 1, instead of an exact algorithm $\mathcal{A}_1$ for the 1-means problem, we can also use a PTAS. Using a PTAS to approximate 1-means up to a $(1+\epsilon)$ factor will only cost an additional $(1+\epsilon)$ factor in our cost analysis. As a result, we get the same approximation and accuracy guarantees if we replace ϵ with $\epsilon/3$.

Algorithm 1 makes $O((k \log n) \cdot m(Q, \epsilon^4, \delta/(k \log n)))$ queries to the oracle $f_{\mathcal{O}}$ in total. Recall that simulating an oracle query to $f_{\mathcal{O}}$ takes $O(k)$ same-cluster queries. Therefore, the total number of same-cluster queries is $O((k^2 \log n) \cdot m(Q, \epsilon^4, \delta/(k \log n)))$.

Our definition of a learning algorithm in Section 2.2 has nothing to do with whether the input is a k-means instance or a k-median instance, which is similar to k-means except that the cost of a cluster C with respect to a center q is defined as $\operatorname{cost}(C, q) := \sum_{p \in C} \mathrm{d}(p, q)$. In fact, it applies to any similar clustering scenario where the cost is defined in terms of the ℓ'th power ($\ell > 0$) of distances instead of squared distances. The analysis of Algorithm 1 can be adapted to any fixed ℓ once we have a suitable triangle inequality. (For example, in k-means (i.e., when $\ell = 2$), we use $(a+b)^2 \le (1+\epsilon)a^2 + (1+1/\epsilon)b^2$. When $\ell \le 1$, we can simply use the trivial triangle inequality $(a+b)^\ell \le a^\ell + b^\ell$.) Thus, for such clustering problems, Algorithm 1, with a slight modification on choice of radii in Step 4 and a little adjustment to the parameter γ, will give the same guarantees. Hence, we have the following theorem which is the formal version of Theorem 1. The proof follows from the analysis of Algorithm 1.

▶ **Theorem 6.** *Let K be a family of k-means (k-median) instances. Suppose that K is learnable with sample complexity $m(Q, \epsilon, \delta)$ using a zero sample error, non-inventive learning*

algorithm $\mathcal{A}_L$. Let $\mathcal{A}_\alpha$ be a constant-factor approximation algorithm, and let $\mathcal{A}_1$ be a PTAS for the 1-means (1-median) problem. There exists a polynomial-time algorithm that, given an instance $(P, Q, \mathrm{d}) \in K$, oracle access to same-cluster queries for some fixed optimal clustering $\mathcal{O}$, and parameters $(\epsilon, \delta) \in (0,1)^2$, outputs a clustering that, with probability at least $(1-\delta)$, is $(1-\epsilon)$-accurate with respect to $\mathcal{O}$, and simultaneously has a cost of at most $(1+\epsilon)OPT$. The algorithm uses $\mathcal{A}_L$, $\mathcal{A}_\alpha$, and $\mathcal{A}_1$ as subroutines. The number of same-cluster queries made by the algorithm is

1. $O((k^2 \log |P|) \cdot m(Q, \epsilon^4, \delta/(k \log |P|)))$ *for the k-means setting and*
2. $O((k^2 \log |P|) \cdot m(Q, \epsilon^2, \delta/(k \log |P|)))$ *for the k-median setting.*

For k-means and k-median instances in Euclidean space and those in finite metric spaces, there exist several constant-factor approximation algorithms (for example, Ahmadian et al. [1] and Kanungo et al. [13]). Solving the 1-means problem in Euclidean space is straightforward: The solution to $\operatorname{argmin}_{q \in \mathbb{R}^r} \operatorname{cost}(P, q)$ is simply $q = (\sum_{p \in P} p)/|P|$. For the k-median problem in Euclidean space, the problem of 1-median does not have an exact algorithm but several PTASes exist (for example, Cohen et al. [6]). In a finite metric space, to solve $\operatorname{argmin}_{q \in Q} \operatorname{cost}(P, q)$, we can simply try all possible $q \in Q$ in polynomial time, and this holds for the k-median setting as well. Thus, for Euclidean and finite metric space k-means and k-median instances that have no boundary points, Theorem 6, together with Theorem 4 and Theorem 5, gives efficient algorithms for $(1+\epsilon)$-approximate, $(1-\epsilon)$-accurate semi-supervised clustering.

4 Removing the Dependency on Problem Size in the Query Complexity for Euclidean k-Means

For the family of Euclidean k-means instances, the query complexity of Algorithm 1 suffers from a $\tilde{O}(\log n)$ dependency (where n is the number of points in the input k-means instance, and $\tilde{O}$ hides $\operatorname{poly}(\log \log n)$ factors) due to the repeated use of the learning algorithm $\mathcal{A}_L$. Specifically, we run $\mathcal{A}_L$ with a failure probability of $\delta/(k \log n)$, $O(\log n)$ times per cluster. Note that the sample complexity of $\mathcal{A}_L$ itself, in the case of Euclidean k-means instances, does not have this dependency.

In this section, we show that we can avoid this dependency on n using a slightly more involved algorithm at the cost of increasing the query complexity by an extra $\operatorname{poly}(k)$ factor. Nevertheless, this algorithm has superior performance when the size of the input instance (i.e., the number of points) is very large (when $\log n = \Omega(k^{10})$ for example).

Recall that, for a set $C \subset \mathbb{R}^r$, $\operatorname{cost}(C, y)$ is minimized when y is the centroid of C, denoted by $\mu(C) = (\sum_{x \in C} x)/|C|$. Define the fractional size of an optimal cluster O_i as the fraction of points that belong to O_i, i.e., the ratio $|O_i|/n$. Suppose we only want to get a good approximation for the cost, and that we know that all the clusters in the target solution have sufficiently large fractional sizes. In this case, naive uniform sampling will likely pick a large number of samples from each of the clusters. This observation, together with Lemma 7, allows us to approximate the centroid and the cost of each cluster to any given accuracy.

▶ **Lemma 7** (Lemma 1 and Lemma 2 of Inaba et al. [12]). *Let $(\epsilon, \delta) \in (0,1)^2$, let $m \geq 1/(\epsilon\delta)$ be a positive integer, and let $S = \{p_1, \ldots, p_m\}$ be a multiset of m i.i.d. samples from the uniform distribution over some finite set $C \subset \mathbb{R}^r$. With probability at least $(1-\delta)$, $\mathrm{d}^2(\mu(S), \mu(C)) \leq \epsilon \cdot \operatorname{cost}(C, \mu(C))/|C|$ and $\operatorname{cost}(C, \mu(S)) \leq (1+\epsilon) \operatorname{cost}(C, \mu(C))$.*

However, the above approach fails when some clusters in the optimal target solution contribute significantly to the cost, but have small fractional sizes (that is because uniform

sampling is not guaranteed to pick sufficient numbers of samples from the small clusters). Ailon et al. [2] circumvented this issue with an algorithm that iteratively approximates the centers of the clusters using a distance-based probability distribution (D^2-sampling). We will refer to their algorithm as $\mathcal{A}^*$.

Note that when it comes to accuracy, we can totally disregard clusters with small fractional sizes; we only have to correctly label a sufficiently large fraction of the points in large clusters. With this intuition, we present the outline of our algorithm.

Let $(P, \mathbb{R}^r, \mathrm{d})$ be a k-means instance in Euclidean space that has no boundary points. For simplicity, we refer to the instance $(P, \mathbb{R}^r, \mathrm{d})$ by just P where possible, as for Euclidean k-means, the other two parameters are fixed. We start with a naive uniform sampling step that gives a good approximation for the centers of large clusters. Starting with these centers, we run a slightly modified version of algorithm $\mathcal{A}^*$ to approximate the centers of the remaining small clusters. Thus, at this point, we have a clustering with a good cost and we know which clusters are large. We now run the learning algorithm $\mathcal{A}_L$ on input P and obtain a labeling of the points. For each point, we assign its final label based on (1) the label assigned to it by the learning algorithm $\mathcal{A}_L$, and (2) its proximity to large cluster centers. In particular, if the output of $\mathcal{A}_L$ decides that a point p should be in some large cluster i, and if p is sufficiently close to the approximate center for cluster i, we label it according to the learning output; otherwise, we label it according to its nearest approximate center. We show that this approach retains a cost that is close to the cost of the clustering output by $\mathcal{A}^*$. The accuracy guarantee comes from the facts that a large fraction of the points are sufficiently close to the centers of large clusters, and that $\mathcal{A}_L$ labels most of them correctly with a good probability.

We now review the key properties of algorithm $\mathcal{A}^*$ (the algorithm of Ailon et al. [2]). Let $0 < \epsilon < 1$. We say a k-means instance P is (k, ϵ)-irreducible if no $(k-1)$-means clustering gives an $(1+\epsilon)$-approximation for the k-means problem, i.e., if OPT^k denotes the optimal k-means cost of P, then P is (k, ϵ)-irreducible if $OPT^{k-1} > (1+\epsilon)OPT^k$. Suppose that P is (k, ϵ)-irreducible. Let $\mathcal{O} = \{O_1, \ldots, O_k\}$ be the target optimal clustering, and let $o_1, \ldots, o_k$ be the respective centers. Let $C_i = \{c_1, \ldots, c_i\}$ denote a set of i centers and let $Z(i)$ denote the following statement: There exists a set of i distinct indices $j_1, \ldots, j_i$ such that, for all $r \in [i]$, $\mathrm{cost}(O_{j_r}, c_r) \leq (1 + \epsilon/16)\,\mathrm{cost}(O_{j_r}, o_{j_r})$. To put it differently, $Z(i)$ says that C_i is a set of good candidate centers for i-many distinct clusters in the target optimal solution. Assuming P is (k, ϵ)-irreducible, the algorithm $\mathcal{A}^*$ yields a method to incrementally construct sets $C_1, \ldots, C_k$ (i.e., $C_{i+1} = C_i \cup \{c_{i+1}\}$) such that, conditioned on $Z(i)$ being true, $Z(i+1)$ is true with probability at least $(1 - 1/k)$. Now suppose that P is $(k, \epsilon/(4k))$-irreducible. Then $\mathcal{A}^*$ gives a $(1+\epsilon/(4 \cdot 16k))$-approximation for k-means with probability at least $(1-1/k)^k \geq 1/4$. Otherwise, $\mathcal{A}^*$ gives a $(1+\epsilon/(4 \cdot 16k))$-approximation for the i-means problem for some $i < k$, where i is the largest integer such that P is $(i, \epsilon/4k)$-irreducible. In the latter case, it will give a $(1 + \epsilon/(4 \cdot 16k))(1 + \epsilon/(4k))^{k-i}$-approximation with probability at least $1/4$. In either case, the output of $\mathcal{A}^*$ is a $(1+\epsilon)$-approximation.

In our algorithm, we first find the centers of large clusters using uniform sampling, and then run $\mathcal{A}^*$ to find the remaining centers. This allows us to know which clusters are large, which is a crucial information needed for the final labeling. Suppose that in the target optimal solution we have $k_0 \leq k$ clusters whose fractional sizes are at least ϵ/k. Note that k_0 is at least 1 due to the Pigeonhole Principle, since at least one cluster should have a fractional size of at least $1/k > \epsilon/k$. By Lemma 7, using uniform sampling, we can approximate the centroid of each of these large clusters with a good accuracy. Hence, we can have a set C_{k_0} of k_0 centers such that $Z(k_0)$ is true with probability $(1 - \delta)$. Afterwards, we use $\mathcal{A}^*$ to

Algorithm 2: Algorithm whose query complexity is independent of n

Input : Point set $P \subset \mathbb{R}^r$, the oracle access to $f_{\mathcal{O}}$, parameter k, accuracy parameter ϵ, failure probability δ, and algorithms $\mathcal{A}_{cost}$ and $\mathcal{A}_L$.

Output : The clustering $\hat{\mathcal{O}} = \{\hat{O}_1, \ldots, \hat{O}_k\}$ defined by the labeling $f_{\hat{\mathcal{O}}} : P \to [k]$ computed below. For each $i \in [k]$, the respective cluster center $\hat{o}_i$ is the centroid of $\hat{O}_i$.

1 Draw $Q_1(k, \epsilon, \delta)$ samples from P independently and uniformly at random, and query $f_{\mathcal{O}}$ to get their true cluster labels in $\mathcal{O}$. Denote the set of sampled points by S, and for all $i \in [k]$, denote the set of sampled points that belong to class i by S_i.

2 Let k' be the number of distinct cluster labels with more than $(\epsilon/(2k))Q_1(k, \epsilon, \delta)$ samples. Let $C_{k'} := \{\mu(S_i) : |S_i| > (\epsilon/(2k))Q_1(k, \epsilon, \delta)\}$. Without loss of generality, assume that the class labels for centers in $C_{k'}$ are $1, \ldots, k'$.

3 Run the algorithm $\mathcal{A}_{cost}$, starting from $C_{k'}$ as the partial set of centers. This takes $Q_2(k, \epsilon, \delta)$ more queries. Let $C_k = \{c_1, \ldots, c_k\}$ be the output, and let OPT^* be the cost of the clustering obtained by assigning each point to its nearest c_i.

4 Use the PAC learning algorithm $\mathcal{A}_L$ on $Q_3(k, r, \epsilon^4/k, \delta)$ uniform i.i.d. samples from P to learn a classifier for the k classes that is $(1 - \epsilon^4/k)$-accurate with probability at least $(1 - \delta)$. Let $H_1, \ldots, H_k$ be the sets of points that are labeled $1, \ldots, k$ respectively by the classifier.

5 Output the clustering $\hat{\mathcal{O}}$ defined by the following labeling function: for each $i \in [k']$ and $p \in H_i$ such that $\mathrm{d}^2(p, c_i) \leq kOPT^*/(n\epsilon^3)$, set $f_{\hat{\mathcal{O}}}(p) = i$. For any other point p, set $f_{\hat{\mathcal{O}}}(p) = i$ if the nearest cluster center to p in C_k is c_i.

incrementally construct $C_{k_0+1}, \ldots, C_k$. Conditioned on $Z(k_0)$ being true, the output C_k will be a $(1 + \epsilon)$-approximation with probability $(1 - 1/k)^{k-k_0} \geq (1 - 1/k)^k \geq 1/4$ for $k \geq 2$. However, by independently running this incremental construction $O(\log(1/\delta))$ times and choosing the set of centers with the minimum total cost, we can boost this probability to $(1 - \delta)$. This observation gives the following generalization of Theorem 10 of Ailon et al. [2].

▶ **Theorem 8.** *Consider a Euclidean k-means instance $(P, \mathbb{R}^r, \mathrm{d})$. Let $O_1, \ldots, O_k$ be a fixed optimal clustering with respective centers $o_1, \ldots, o_k$. Let $k_0 \leq k$ and let $C_{k_0} = \{c_1, \ldots, c_{k_0}\}$ be a set of points such that, with probability at least p_0, $\mathrm{cost}(O_i, c_i) \leq (1 + \epsilon/(64k))\,\mathrm{cost}(O_i, o_i)$ for all $i \in [k_0]$. There exists an algorithm $\mathcal{A}_{cost}$ that, given P, C_{k_0}, and parameters $(\epsilon, \delta) \in (0, 1)^2$ as input, outputs a set of centers $C_k = C_{k_0} \cup \{c_{k_0+1}, \ldots, c_k\}$ such that $\sum_{i\in[k]} \mathrm{cost}(O_i, c_i) \leq (1 + \epsilon) \sum_{i\in[k]} \mathrm{cost}(O_i, o_i)$ with probability at least $p_0(1 - \delta)$. Moreover, $\mathcal{A}_{cost}$ uses $O((k^9/\epsilon^4)\log(1/\delta))$ same-cluster queries and runs in time $O(nr(k^9/\epsilon^4)\log(1/\delta))$.*

Theorem 8 implies a method to get a good approximation for the cost that also reveals which clusters are large. Specifically, we first perform uniform sampling over the whole set P and approximate the centers of the large clusters. If we get a sufficient number of samples, the approximate centers will satisfy the precondition of Theorem 8 with a good probability. Thus, using algorithm $\mathcal{A}_{cost}$ from Theorem 8, we get the desired approximation for the cost. What remains now is to use PAC learning and to appropriately label the points according to the learning outcome.

We present the pseudo-code of our algorithm in Algorithm 2, where we use the algorithm $\mathcal{A}_{cost}$ from Theorem 8 and the learning algorithm $\mathcal{A}_L$ guaranteed by Theorem 4. In Algorithm 2, $Q_1(k, \epsilon, \delta) = 256k^3/(\epsilon^2\delta)$ is the number of samples needed to ensure that we pick a sufficient number of samples from each of the clusters with fractional sizes of at least ϵ/k, $Q_2(k, \epsilon, \delta)$ is the sample complexity of the algorithm $\mathcal{A}_{cost}$, and $Q_3(k, r, \epsilon, \delta) = m(\mathbb{R}^r, \epsilon, \delta)$ is the sample complexity of the learning algorithm $\mathcal{A}_L$ for an error ϵ and a failure probability δ. As with Algorithm 1, the center that a point is assigned to in the final output may not be the closest center to that point.

With probability at least $(1 - \delta)$, the output of Algorithm 2 is $(1 + \epsilon)$-approximate and $(1 - \epsilon)$-accurate (refer to the full version for the complete analysis). As for the claim on the

query complexity, recall that we only need $O(k)$ same-cluster queries per single $f_{\mathcal{O}}$ query, and Algorithm 2 makes a total number of $Q_1(k,\epsilon,\delta) + Q_2(k,\epsilon,\delta) + Q_3(k,r,\epsilon^4/k,\delta)$ queries to $f_{\mathcal{O}}$. This observation together with the analysis of Algorithm 2 proves Theorem 2. We remark that the query complexity $Q_3(k,r,\epsilon^4/k,\delta)$ for learning Euclidean k-means instances is independent of P.

References

1 Sara Ahmadian, Ashkan Norouzi-Fard, Ola Svensson, and Justin Ward. Better guarantees for k-means and euclidean k-median by primal-dual algorithms. *2017 IEEE 58th Annual Symposium on Foundations of Computer Science (FOCS)*, pages 61–72, 2017.

2 Nir Ailon, Anup Bhattacharya, Ragesh Jaiswal, and Amit Kumar. Approximate clustering with same-cluster queries. In *9th Innovations in Theoretical Computer Science Conference, ITCS 2018, January 11-14, 2018, Cambridge, MA, USA*, pages 40:1–40:21, 2018. doi:10.4230/LIPIcs.ITCS.2018.40.

3 Hassan Ashtiani, Shrinu Kushagra, and Shai Ben-David. Clustering with same-cluster queries. In *Proceedings of the 30th International Conference on Neural Information Processing Systems*, NIPS'16, pages 3224–3232, USA, 2016. Curran Associates Inc. URL: http://dl.acm.org/citation.cfm?id=3157382.3157458.

4 P. Awasthi, A. Blum, and O. Sheffet. Stability yields a PTAS for k-median and k-means clustering. In *2010 IEEE 51st Annual Symposium on Foundations of Computer Science*, pages 309–318, Oct 2010. doi:10.1109/FOCS.2010.36.

5 Maria-Florina Balcan, Avrim Blum, and Anupam Gupta. Clustering under approximation stability. *J. ACM*, 60(2):8:1–8:34, 2013. doi:10.1145/2450142.2450144.

6 Michael B. Cohen, Yin Tat Lee, Gary Miller, Jakub Pachocki, and Aaron Sidford. Geometric median in nearly linear time. In *Proceedings of the Forty-eighth Annual ACM Symposium on Theory of Computing*, STOC '16, pages 9–21, New York, NY, USA, 2016. ACM. doi:10.1145/2897518.2897647.

7 Vincent Cohen-Addad, Philip N. Klein, and Claire Mathieu. Local search yields approximation schemes for k-means and k-median in euclidean and minor-free metrics. In *IEEE 57th Annual Symposium on Foundations of Computer Science, FOCS 2016, 9-11 October 2016, Hyatt Regency, New Brunswick, New Jersey, USA*, pages 353–364, 2016.

8 S. Dasgupta. *The Hardness of K-means Clustering.* Technical report (University of California, San Diego. Department of Computer Science and Engineering). Department of Computer Science and Engineering, University of California, San Diego, 2008. URL: https://books.google.ch/books?id=riJuAQAACAAJ.

9 Dan Feldman, Morteza Monemizadeh, and Christian Sohler. A PTAS for k-means clustering based on weak coresets. In *Proceedings of the Twenty-third Annual Symposium on Computational Geometry*, SCG '07, pages 11–18, New York, NY, USA, 2007. ACM. doi:10.1145/1247069.1247072.

10 Zachary Friggstad, Mohsen Rezapour, and Mohammad R. Salavatipour. Local search yields a PTAS for k-means in doubling metrics. In *IEEE 57th Annual Symposium on Foundations of Computer Science, FOCS 2016, 9-11 October 2016, Hyatt Regency, New Brunswick, New Jersey, USA*, pages 365–374, 2016.

11 Sariel Har-Peled and Soham Mazumdar. On coresets for k-means and k-median clustering. In *Proceedings of the thirty-sixth annual ACM symposium on Theory of computing*, pages 291–300. ACM, 2004.

12 Mary Inaba, Naoki Katoh, and Hiroshi Imai. Applications of weighted voronoi diagrams and randomization to variance-based k-clustering: (extended abstract). In *Proceedings of*

the Tenth Annual Symposium on Computational Geometry, SCG '94, pages 332–339, New York, NY, USA, 1994. ACM. doi:10.1145/177424.178042.

13 Tapas Kanungo, David M. Mount, Nathan S. Netanyahu, Christine D. Piatko, Ruth Silverman, and Angela Y. Wu. A local search approximation algorithm for k-means clustering. In *Proceedings of the Eighteenth Annual Symposium on Computational Geometry*, SCG '02, pages 10–18, New York, NY, USA, 2002. ACM. doi:10.1145/513400.513402.

14 Euiwoong Lee, Melanie Schmidt, and John Wright. Improved and simplified inapproximability for k-means. *Inf. Process. Lett.*, 120:40–43, 2017.