Combinatorial Feature Selection Problems

Moses Charikar* Venkatesan Guruswami† Ravi Kumar‡ Sridhar Rajagopalan§
Amit Sahai¶

Abstract

Motivated by frequently recurring themes in information retrieval and related disciplines, we define a genre of problems called *combinatorial feature selection* problems. Given a set S of multidimensional objects, the goal is to select a subset K of relevant dimensions (or features) such that some desired property Π holds for the set S restricted to K. Depending on Π, the goal could be to either maximize or minimize the size of the subset K. Several well-studied feature selection problems can be cast in this form. We study the problems in this class derived from several natural and interesting properties Π, including variants of the classical p-center problem as well as problems akin to determining the VC-dimension of a set system. Our main contribution is a theoretical framework for studying combinatorial feature selection, providing (in most cases essentially tight) approximation algorithms and hardness results for several instances of these problems.

1 Introduction

In the simplest *vector-space model* for text, a document is viewed as a set of words and phrases (more generally, *features*) which occur in it [27]. The cardinality of this feature set can get daunting—for instance, on the web, the number of different words used (even when restricted to pages in English) is in the millions. The large number of features present a significant engineering challenge for data mining, document classification, or clustering applications and, simultaneously, pose the even more significant risk of overfitting.[1]

The standard approach to alleviate many of these problems is to restrict attention to a carefully chosen *subset* of the feature set. This is called *feature selection*. This provides many benefits: (i) the processing and data management tasks get significantly more tractable, (ii) the risk of overfitting is largely avoided, and (iii) noisy features are eliminated. The obvious question which arises in this context is: which set of features do we retain and which ones do we discard? Posed in this generality, however, there is no hope of obtaining a universal answer to this question and the answer depends on the intent of the original data processing problem.

For example, suppose we are given a set of distinct objects with various distinguishing attributes, and say they are represented by vectors in some high-dimensional space. An interesting goal then is to pick a small

*Computer Science Department, Stanford University, CA 94305. Research supported by the Pierre and Christine Lamond Fellowship, NSF Grant IIS-9811904 and NSF Award CCR-9357849, with matching funds from IBM, Mitsubishi, Schlumberger Foundation, Shell Foundation, and Xerox Corporation. Most of this work was done while the author was visiting IBM Almaden Research Center. moses@theory.stanford.edu

†MIT Laboratory for Computer Science, 545 Technology Square, Cambridge, MA 01239. Most of this work was done while the author was visiting IBM Almaden Research Center. venkat@theory.lcs.mit.edu

‡IBM Almaden Research Center, 650 Harry Road, San Jose, CA95120. ravi@almaden.ibm.com

§IBM Almaden Research Center, 650 Harry Road, San Jose, CA95120. sridhar@almaden.ibm.com

¶MIT Laboratory for Computer Science, 545 Technology Square, Cambridge, MA 01239. Supported in part by a DOD NDSEG Fellowship. Most of this work was done while the author was visiting IBM Almaden Research Center. amits@theory.lcs.mit.edu.

[1] In fact, irrelevant features (e.g., stopwords) can and do mask underlying patterns in text data.

subset of relevant dimensions which still suffice to "tell apart" all the objects; this genre of problems are known as *dimension reduction* problems, since we are obtaining the representation of the objects in a lower-dimensional space that still "explains" their properties. A different scenario in which feature selection arises is when we have an underlying set of points in high-dimensional space which we know a priori to "cluster well", but, due to the presence of "noisy" dimensions the clustering is destroyed when all dimensions are considered together. The aim here is to throw out a set of noisy dimensions so that the data clusters well in all the remaining dimensions; we refer to this as the *hidden clusters* problem.

Thus, feature selection problems all come equipped with some underlying property on sets of vectors, and the goal is to pick a maximum or minimum number of dimensions (depending upon the application) such that the property holds on the chosen set of dimensions.

Our contributions. In this paper, we provide a unified theoretical framework for studying combinatorial feature selection problems in general. Our framework captures both the dimension reduction and clustering problems discussed above among other combinatorial feature selection problems. These problems turn out to be extremely hard in even the simplest of set-ups—often the underlying property we wish to satisfy is itself non-trivial and we have to deal with the non-obvious task of which subset of dimensions to pick.

We consider several specific instances of problems which fall within our framework, and provide (in most cases essentially tight) approximation algorithms and hardness results. The precise statement of our results can be found in Section 2, we just mention one example here. Consider the *hidden cluster* problem with the L_∞ metric to measure cluster radius and where the data is a priori "known" to cluster into p sets for a constant p (this is just the hidden cluster analogue of the classical p-center problem). We give a polynomial time algorithm for this problem that retains the maximum number of dimensions with a small (factor 3) slack in the cluster radius. On the other hand, obtaining any reasonable ($n^{1-\epsilon}$) factor approximation on the number of dimensions with a better than factor 2 slack in the radius turns out to be NP-hard, even when there are known to be only two clusters! This should give some indication of the non-trivial nature of these problems.

Combinatorial vs. affine versions. In a combinatorial feature selection problem, the selected subspace is defined by choosing some of the original dimensions and discarding the rest. In contrast, "affine" versions allow the choice of any affine subspace. Though the affine version is more studied (at least in the classical theory literature), the combinatorial version is interesting in its own right and has several practical merits, the most compelling one being that the resulting (low-dimensional) space is "interpretable", i.e., the selected dimensions (features) have a real meaning in the context of the original data set and consequently to the user. In the context of data mining and searching, interpretability is a significant concern as studies have shown that it improves the utility and understanding of search and classification results by human subjects, especially with visualization techniques [13]. Even a simple linear combination of many dimensions may be hard to interpret [2].

Other practical considerations include the following. (i) The cost of applying combinatorial feature selection to the given set of vectors is significantly less than the cost of applying an affine feature selection (which involves a linear transform). In practice, this turns out to be an important issue whenever clustering efficiency and scalability becomes more important than (incremental benefits in) classifier efficiency and predictive accuracy [8]. (ii) It might seem that the increased flexibility provided by allowing affine subspaces as features could result in significantly improved classification and clustering accuracy. Surprisingly, several authors provide strong evidence to the contrary [5, 20, 11]. They have successfully used combinatorial approaches to feature selection and shown that the resulting *gains* in both the efficiency and quality of clustering and classification algorithms are significant.

Related work. Feature selection problems have received extensive attention in the more classical affine setting. For instance, see the work of Johnson and Lindenstrauss [18] for $\mathbb{R}^n$, and others [4, 22, 3] for more general metrics. One of the widely used dimension reduction techniques is the singular value decomposition

(SVD) [10, 24]. While the generic goal is to find a low dimensional representation of the original space, the new dimensions (i.e., features), however, are not restricted to be a subset of the original features. In the case of Johnson and Lindenstrauss and SVD, the new features turn out to be affine linear combinations of the original features; in other cases, they are derived in a somewhat more complicated manner. The resulting dimension reduction problems have been analyzed (cf. [22, 3]) and are provably useful in many contexts (cf. [28, 9]).

Koller and Sahami [19] study feature selection in an information-theoretic context. They propose that choosing a subset of dimensions that minimizes the Kullback-Liebler divergence (or the cross-entropy) between the distribution on the classes given the data and the projected data. While the strategy is not directly implementable, it becomes tractable when a Bernoulli model is posited on the data. While Bernoulli models are popular and seem to perform well for most classification and clustering problems, many real world data sets, including text corpora, are known to adhere to the Zipf [30] statistic.

Another modern approach to feature selection is the "wrapper" scheme [17]. In this, an exhaustive enumeration of all subsets of the feature space is evaluated by training on a training corpus and testing against a reserved test corpus. This method tends to be prohibitively expensive in practice, especially when the number of features is large—though it does not need to make an assumption about the data distribution.

A technique proposed in [8] is to use a wrapper scheme similar to [17], but only consider a linear number of subsets of the feature space. This is done by ordering the dimensions according to some desirability criterion, and then considering only prefixes of the ordering. A similar idea is used in [5]. While this reduces the combinatorial explosion, it effectively assumes some form of independence among the attribute dimensions.

2 A framework for feature selection problems

Let $S = \{x_i : 1 \leq i \leq m\}$ denote a set of vectors (also referred to as points), where each $x_i \in \mathcal{M}_1 \times \cdots \times \mathcal{M}_n$, where $\mathcal{M}_j$ is a metric space equipped with metric $\text{dist}_j(\cdot, \cdot)$. Throughout the paper n will denote the number of dimensions and m the number of points x_i. Let $x_{i|K}$ denote the projection of x_i onto the subspace indexed by the dimensions in K and let $S_{|K} \stackrel{\text{def}}{=} \{x_{i|K}\}$. Feature selection corresponds to selecting a subset $K \subseteq \{1, \ldots, n\}$ such that $S_{|K}$ has some "good properties." In an optimization setting, two complementary flavors of the feature selection problem can be defined—*subspace selection* and *dimension reduction*.

Let $\Pi(S)$ be some property of a set of vectors S. In *subspace selection*, we are given an S which does not satisfy Π, and we want to find the largest K for which $\Pi(S_{|K})$ holds (or possibly, holds in some relaxed manner). On the other hand, in *dimension reduction*, we are given an S such that $\Pi(S)$ holds and we wish to find the smallest set K, such that $\Pi(S_{|K})$ holds (or possibly, holds in some relaxed manner). Both these versions have the interpretation of retaining a set of relevant dimensions so that the vectors have the desired properties when projected onto those dimensions.

In a typical feature selection problem, the property Π is parametrized to indicate how well the property is satisfied, e.g. the maximum radius r of a clustering, or the number ℓ of distinct points. The relevance of a subset K of dimensions is judged by how well property Π is satisfied by $S_{|K}$. If property Π is such that it is made easier to satisfy by discarding dimensions, the corresponding feature selection problem is a *subspace selection* problem, i.e., we try to *maximize* the number of dimensions in our subset K. On the other hand, if property Π is such that it is made easier to satisfy by adding dimensions, the corresponding feature selection problem is a *dimension reduction* problem, i.e., we try to *minimize* the number of dimensions in K.

Several interesting problems fall in the abstract framework of feature selection problems described above. First, we look at subspace selections problems related to *clustering*. Suppose, the input points contain a "hidden clustering," in that one can pick a subset of dimensions such that the points, when projected onto this subset of dimensions, can be clustered into a small number of groups such that distances between points in the same cluster are small. In particular we will be interested in the L_1 and L_∞ norms:

$$\begin{aligned}\mathrm{dist}^{(1)}_{|K}(x,y) &= \sum_{j\in K}\mathrm{dist}_j(x,y)\ ,\\ \mathrm{dist}^{(\infty)}_{|K}(x,y) &= \max_{j\in K}\mathrm{dist}_j(x,y)\ .\end{aligned}$$

Hidden clusters problems. We study clustering with the min-max objective, i.e. minimizing the maximum distance of a point to its cluster center. Our hidden cluster problems can be viewed as multidimensional analogs of the classic p-center problem (cf. [16, 12]).

The L_∞ *hidden cluster* problem is the following: given *radius* r and ℓ, find ℓ centers $C = \{c_1, \ldots, c_\ell\}$, an assignment of points to centers $\sigma : \{x_i\} \to C$, and a subset of dimensions K such that $\mathrm{dist}^{(\infty)}_{|K}(x_i, \sigma(x_i)) \le r$ for all points x and $|K|$ is maximized. The L_1 *hidden cluster* problem is similar, except that the radius requirement is $\mathrm{dist}^{(1)}_{|K}(x_i, \sigma(x_i)) \le r$ for all points x_i.

For the hidden clusters problem, we will be interested in *bicriteria approximation algorithms*, i.e., algorithms that approximate both the radius and the number of dimensions. Suppose the optimal solution uses radius r and k dimensions. An (α, β)-approximation algorithm is one that guarantees a solution that has radius at most αr and returns at least k/β dimensions. It will be convenient to state our results as bicriteria approximations.

Dimension reduction problems for Boolean vectors. When S is restricted to Boolean vectors, a number of interesting dimension reduction problems arise. We now provide several instances that we consider in this paper.

(i) **Entropy maximization problem**: Find K such that the entropy of the random variable with distribution $U(S_{|K})$ is maximized. Here $U(\cdot)$ denotes the uniform distribution. Suppose there are q distinct elements $y_1, \ldots, y_q$ in $S_{|K}$ and m_i elements take the value y_i; $\sum_{i=1}^{q} m_i = m$. The entropy $H(U(S_{|K})) = -\sum_{i=1}^{q} p_i \lg p_i$, where $p_i = m_i/m$.

The entropy objective function encourages us to find K such that $S_{|K}$ has a large number of distinct elements; moreover it favors a somewhat equal distribution of rows amongst the various distinct elements. This corresponds to dividing the vectors into a large number of equivalence classes so that the distribution of vectors amongst equivalence classes is not very skewed. Note that if $H(U(S_{|K})) \ge \lg \ell$, then $S_{|K}$ must have at least ℓ distinct elements. A heuristic for a similar objective function is studied in [8]. We consider the problem of maximizing the entropy given a bound k on K, as well as the dual problem of minimizing $|K|$ such that $H(S_{|K}) \ge e$ where e is given.

(ii) **Distinct vectors problem**: Given S, a collection of distinct vectors, find the smallest K such that $S_{|K}$ are still distinct.

(iii) **Maximum distinct points problem**: Maximize the number of distinct elements in $S_{|K}$, given an upper bound k on $|K|$—considerations similar to this are motivated by the study of VC-dimension of set systems [25]. We also consider the dual problem where given ℓ, we wish to minimize K so that $S_{|K}$ has at least ℓ distinct points.

Our results. We give a deterministic $(3, 1)$-approximation algorithm (Section 3.1) for the L_∞ hidden cluster problem and a randomized (quasi-polynomial) $(O(\lg m), 1 + \epsilon)$-approximation algorithm (Section 3.2) for the L_1 version, for any $\epsilon > 0$ when the number of clusters ℓ is constant. Through reductions from CLIQUE and DENSEST SUBGRAPH, we provide evidence that the exponential dependence in the running time on the number of clusters is likely to be inherent (Section 4.1). Furthermore, even for constant number of centers, for any constants $\delta > 0, c > 1$, we show that it is NP-hard to obtain a $(2-\delta, n^{1-\delta})$-approximation algorithm for the L_∞ hidden cluster problem and a $(c, n^{1-\delta})$-approximation algorithm for the L_1 version (Section 4.2).

These highlight the extreme hardness of these problems and the fact that the approximation guarantees of our algorithms are quite close to the best possible.

We give a $(1 - e^{-1})$-approximation algorithm for the entropy maximization problem and a $\lg m$-approximation algorithm for its dual (Section 5.1). We then show a tight relationship between the approximability of the distinct vectors problem and SET COVER (Theorem 20 and also show that the maximum distinct points problem and its dual version are related to DENSEST SUBGRAPH.

3 Algorithms for hidden cluster problems

3.1 L_∞ hidden cluster

We now give a $(3, 1)$-approximation algorithm for the L_∞ hidden cluster problem, i.e., if there is an optimal solution to the L_∞ hidden cluster problem with ℓ clusters of radius r on k "relevant" dimensions, the algorithm will find a solution with ℓ clusters of radius at most $3r$ and will "pick" at least k dimensions. The algorithm runs in polynomial time provided ℓ is constant.

Intuition. The high level idea behind our solutions for both the L_∞ and L_1 hidden cluster problems is the following. Since the number of centers ℓ is a constant, we can assume that we know the set of centers in an optimal solution S^* (as the algorithm can try all possible choices of ℓ centers and output the best solution found). We show that once the centers are known, there is a *small* subset K_{dist} of the set of dimensions K^* picked in the optimal solution S^*, and an efficient way to assign the m points to the centers *only based on dimensions in* K_{dist} (according to a "suitable" criterion) that gives a fairly "accurate" description of the assignment in the optimal solution S^* and thus achieves a good approximation. Since the set of dimensions K_{dist} is "small", the algorithm can search over all choices of K_{dist}, and output the best solution found.

1. For all possible choices of ℓ centers $C = \{c_1, \ldots, c_\ell\}$ and subset K' of dimensions, $|K'| \leq \binom{\ell}{2}$ do:
   (a) For every point x, assign x to a chosen center c_i such that $\text{dist}^{(\infty)}_{|K'}(x, c_i) \leq r$ (ties broken arbitrarily). If no such center exists, fail and go on to next choice of centers and dimensions.
   (b) For point x, let $\sigma(x)$ denote the center that x is assigned to.
   (c) Choose $K_{K',C} = \{j : \forall x, \text{dist}_j(x, \sigma(x)) \leq 3r\}$.
2. Return $\arg\max_{K',C}\{|K_{K',C}|\}$, the largest set generated during the above process.

Figure 1: Algorithm L_∞ hidden cluster

The Algorithm. We will assume that the algorithm knows the optimum radius r.[2] The algorithm appears in Figure 1. The algorithm guesses a set of ℓ centers and a subset of dimensions, and uses the subset of dimensions to assign points to centers. Having come up with an assignment, it then includes all dimensions for which points are close to their assigned centers. It is clear that the algorithm runs in polynomial time for constant ℓ. The running time of the algorithm is dominated by the number of choices of ℓ centers and at most $\binom{\ell}{2}$ dimensions. For each choice of centers and dimensions, the algorithm runs in $O(m\ell^3 + mn)$ time. The overall running time of the algorithm is $O(m(\ell^3 + n)m^\ell n^{\binom{\ell}{2}})$.

[2]If the optimum radius is not known and the number of dimensions k is specified instead, we simply run the algorithm for all values of r (there are only polynomially many possibilities). The solution with the smallest value of r which includes at least k dimensions is returned.

We now analyze the algorithm. Suppose the optimal solution picks centers $C^* = \{c_1, \ldots, c_\ell\}$ and subset K^* of dimensions, $|K^*| = k$. Let r be the radius of the optimum solution. We build a subset K_{sep} of at most $\binom{\ell}{2}$ dimensions as follows. For every pair of centers c_i, c_j, such that $\text{dist}^{(\infty)}_{|K^*}(c_i, c_j) > 2r$, pick a dimension d in K^* such that $\text{dist}_d(c_i, c_j) > 2r$. $K_{\text{sep}} \subseteq K^*$ consists of all the dimensions picked in this way.

Call a run of the algorithm *lucky* when the ℓ centers $c_1, \ldots, c_\ell$ and the subset K_{sep} of dimensions is chosen in the first step.

Lemma 1 *In a lucky run, $\forall x$, if x is assigned to c_i in the optimal solution, then dist$^{(\infty)}_{|K_{\text{sep}}}(x, c_i) \leq r$.*

Proof. Since x is assigned to c_i in the optimal solution, $\text{dist}^{(\infty)}_{|K^*}(x, c_i) \leq r$. Further, since $K_{\text{sep}} \subseteq K^*$, $\text{dist}^{(\infty)}_{|K_{\text{sep}}}(x, c_i) \leq \text{dist}^{(\infty)}_{|K^*}(x, c_i) \leq r$. ■

As a consequence of Lemma 1, the algorithm does not fail when the parameters chosen in the first step correspond to the lucky run. In a lucky run, the following lemma guarantees that if the algorithm assigns a point to a center different from that chosen by the optimal, the two centers are close in the optimal solution.

Lemma 2 *For a lucky run of the algorithm, suppose the optimal solution assigns x to c_i and the above algorithm assigns x to c_j. Then dist$^{(\infty)}_{|K^*}(c_i, c_j) \leq 2r$.*

Proof. Let $c_i \neq c_j$ and assume for contradiction that $\text{dist}^{(\infty)}_{|K^*}(c_i, c_j) > 2r$. By the construction of K_{sep}, there must be one dimension $d \in K_{\text{sep}}$ such that $\text{dist}_d(c_i, c_j) > 2r$. Since x is assigned to c_i in the optimal solution, $\text{dist}^{(\infty)}_{|K^*}(x, c_i) \leq r$ which implies that $\text{dist}_d(x, c_i) \leq r$. From triangle inequality $\text{dist}_d(x, c_j) \geq \text{dist}_d(c_i, c_j) - \text{dist}_d(x, c_i) > r$. Hence $\text{dist}_{|K_{\text{sep}}}(x, c_j) \geq \text{dist}_d(x, c_j) > r$. Therefore, the algorithm could not have assigned x to c_j. Thus, $\text{dist}^{(\infty)}_{|K^*}(c_i, c_j) \leq 2r$. ■

Lemma 3 *In a lucky run, the algorithm returns a solution of radius at most $3r$ and a subset of dimensions that includes K^*.*

Proof. By construction, all dimensions j included in the final solution satisfy $\text{dist}_j(x, \sigma(x)) \leq 3r$. If K is the subset of dimensions picked, then $\text{dist}_{|K}(x, \sigma(x)) \leq 3r$ for all points x. Hence the solution returned has radius at most $3r$.

Consider point x which is assigned to center c_i by the optimal solution. By Lemma 2, if the algorithm assigns x to center c_j then $\text{dist}^{(\infty)}_{K^*}(c_i, c_j) \leq 2r$. Also, since x is assigned to c_i in the optimal solution, $\text{dist}^{(\infty)}_{K^*}(x, c_i) \leq r$. By the triangle inequality, $\text{dist}^{(\infty)}_{K^*}(x, c_j) \leq 3r$. This means that for all dimensions $d \in K^*$, $\text{dist}_d(x, \sigma(x)) \leq 3r$ for all points x. Thus all the dimensions in the optimal solution satisfy the condition checked for in the third step of the algorithm and hence all of them will be included in the solution returned. ■

Since the returned set of dimensions is at least as large as K^*, we obtain the following theorem.

Theorem 4 Algorithm L_∞ hidden cluster *is a $(3, 1)$-approximation algorithm.*

3.2 L_1 hidden cluster

The steps in the algorithm for L_1 hidden cluster are very similar to those in the algorithm for the L_∞ case. The basic idea is to guess a set of ℓ centers and a subset K of dimensions. We use the subset of dimensions to assign the points to centers. Having fixed an assignment, we find a large subset of dimensions such that for this assignment, every point is close to its assigned center. Since we use the L_1 norm for computing distances, the individual steps are more complicated than in the previous algorithm. Firstly, we need to argue that a small subset of dimensions can be used to figure out the assignment of points to centers. In fact, we will use a multi-set of dimensions of size $O(\lg m)$ for every pair of centers. Secondly, in order to find a large subset of dimensions that are good for the assignment produced, we will need to solve a packing integer program (PIP). Since we need to guess $O(\lg m)$ size multi-sets of dimensions to obtain the assignment, our algorithm will run in $n^{O(\lg m)}$ time. Further, since we need to solve a PIP in order to obtain our final solution, we lose a factor of $O(\lg m)$ on the radius.

As before, we will assume that the algorithm knows the optimum radius r. The algorithm is described in Figure 2.

1. For all choices of ℓ centers $c_1, \ldots, c_\ell$ and for all undirected graphs G on the ℓ centers do:
   (a) For every pair of centers c_i, c_j such that G has an edge from c_i to c_j, pick a multi-set K_{ij} of $19 \lg m$ dimensions. Over all choices of K_{ij} do:
      i. For every point x build a directed graph G_x on centers as follows:
      ii. For all $(c_i, c_j) \in G$, compute $\alpha(x, c_i, c_j, K_{ij}) = \sum_{\kappa \in K_{ij}} \frac{\text{dist}_\kappa(x, c_i) - \text{dist}_\kappa(x, c_j)}{\text{dist}_\kappa(c_i, c_j)}$
      iii. Place a directed edge from c_i to c_j in G_x if this quantity is > 0, else place a directed edge from c_j to c_i in G_x.
      iv. For all points x, assign x to any sink (*i.e.* any vertex with no outgoing edges) in the graph G_x. If for some x, G_x does not have a sink, go on to the next choice in Step 1(a).
      v. Let $\sigma(x)$ denote the center that x is assigned to.
      vi. Let K denote the set of dimensions κ such that $\text{dist}_\kappa(x, \sigma(x)) \leq 4r$ for all x.
      vii. Write the following PIP: $\max \sum_{\kappa \in K} y_\kappa$ subject to $\forall x, \sum_{\kappa \in K} y_\kappa \cdot \text{dist}_\kappa(x, \sigma(x)) \leq 4r$ and $y_\kappa \in \{0, 1\}$. (The value of y_κ corresponds to choosing whether or not to include dimension κ in the solution.)
      viii. Solve the LP relaxation of this obtained by relaxing the last constraint to $y_\kappa \in [0, 1]$ to obtain LP^*.
      ix. Obtain an integer solution to the PIP by applying randomized rounding to the fractional LP solution such that the value of the integral solution is $\Omega(LP^*)$ and the packing constraints are violated by a factor of $O(\lg m)$. This gives a subset of dimensions K' consisting of all dimensions κ for which $y_\kappa = 1$.
2. Amongst all solutions generated with radius at most $O(\lg m) \cdot r$, return the solution with the largest number of dimensions.

Figure 2: Algorithm L_1 hidden cluster

We now analyze the algorithm. Let r be the radius, $c_1, \ldots, c_\ell$ be the centers and K^* be the subset of dimensions chosen in the optimal solution. We begin with a technical Lemma that guarantees the existence of a small (i.e., $O(\lg m)$ sized) multi-set of the n dimensions that allows us to *distinguish* between pairs c_i, c_j of centers, i.e., for all points x that are assigned to either c_i or c_j, it allows us to tell correctly whether x is assigned to c_i or to c_j. Formally, for an assignment of points to centers and a pair of centers c_i, c_j, a multi-set K of dimensions is said to be *distinguishing* if $\alpha(x, c_i, c_j, K) = \sum_{\kappa \in K} \frac{\text{dist}_\kappa(x,c_i) - \text{dist}_\kappa(x,c_j)}{\text{dist}_\kappa(c_i,c_j)}$ is > 0 for all points x assigned to c_j and $\alpha(x, c_i, c_j, K) < 0$ for all points x assigned to c_i.

We will need the following version of Hoeffding's bound. (cf. [23]):

Lemma 5 *Let $Y_1, \ldots, Y_n$ be independent and identically distributed random variables over $[-1, 1]$, and* *define* $Y = \sum_{i=1}^{n} Y_i$. *Then* $\Pr[Y - \mathbf{E}[Y] > \delta] \leq \exp(-\delta^2/2n)$.

Lemma 6 *Consider a pair of centers* c_i, c_j *such that* $\mathrm{dist}_{K^*}(c_i, c_j) > 3r$. *Then, for the assignment for points to centers in the optimal solution and the pair of centers* c_i, c_j, *there exists a* $O(\lg m)$ *sized distinguishing multi-set* K_{ij} *of dimensions.*

Proof. We will give a probabilistic proof of existence of a distinguishing multi-set of dimensions. We define a probability distribution on dimensions $\kappa \in K^*$ (note that we do not know the set K^*, but this is okay as we are only demonstrating the *existence* of the desired multi-set). We will pick an $O(\lg m)$ sized multi-set of dimensions by sampling from this distribution with replacement, and show that with positive probability the multi-set of dimensions obtained is distinguishing.

The probability distribution is defined as follows: Choose dimension κ with probability p_κ proportional to $\mathrm{dist}_\kappa(c_i, c_j)$, i.e.,

$$p_\kappa = \frac{\mathrm{dist}_\kappa(c_i, c_j)}{\mathrm{dist}^{(1)}_{K^*}(c_i, c_j)}.$$

Observe that $\mathrm{dist}^{(1)}_{K^*}(c_i, c_j) = \sum_{\kappa \in K^*} \mathrm{dist}_\kappa(c_i, c_j)$. Thus $\sum_{\kappa \in K^*} p_\kappa = 1$. Let $Y(x)$ be a random variable that takes value

$$\alpha(x, c_i, c_j, \kappa) = \frac{\mathrm{dist}_\kappa(x, c_i) - \mathrm{dist}_\kappa(x, c_j)}{\mathrm{dist}_\kappa(c_i, c_j)}$$

with probability p_κ. Note that $Y(x) \in [-1, 1]$ by triangle inequality. Also,

$$\begin{aligned}
\mathbf{E}[Y(x)] &= \sum_{\kappa \in K^*} p_\kappa \cdot \alpha(x, c_i, c_j, \kappa) \\
&= \sum_{\kappa \in K^*} \frac{\mathrm{dist}_\kappa(c_i, c_j)}{\mathrm{dist}^{(1)}_{K^*}(c_i, c_j)} \cdot \frac{\mathrm{dist}_\kappa(x, c_i) - \mathrm{dist}_\kappa(x, c_j)}{\mathrm{dist}_\kappa(c_i, c_j)} \\
&= \sum_{\kappa \in K^*} \frac{\mathrm{dist}_\kappa(x, c_i) - \mathrm{dist}_\kappa(x, c_j)}{\mathrm{dist}^{(1)}_{K^*}(c_i, c_j)} \\
&= \frac{\mathrm{dist}^{(1)}_{K^*}(x, c_i) - \mathrm{dist}^{(1)}_{K^*}(x, c_j)}{\mathrm{dist}^{(1)}_{K^*}(c_i, c_j)}
\end{aligned}$$

Suppose x is assigned to c_i in the optimal solution. Then $\mathrm{dist}_{K^*}(x, c_i) \leq r$. Also, $\mathrm{dist}_{K^*}(c_i, c_j) > 3r$ (by the hypothesis). By triangle inequality, we can upper bound $\mathbf{E}[Y(x)]$ by:

$$\begin{aligned}
&\frac{\mathrm{dist}^{(1)}_{K^*}(x, c_i) + (\mathrm{dist}^{(1)}_{K^*}(x, c_i) - \mathrm{dist}^{(1)}_{K^*}(c_i, c_j))}{\mathrm{dist}^{(1)}_{K^*}(c_i, c_j)} \\
&= 2 \cdot \frac{\mathrm{dist}^{(1)}_{K^*}(x, c_i)}{\mathrm{dist}^{(1)}_{K^*}(c_i, c_j)} - 1 < 2 \cdot \frac{1}{3} - 1 = -\frac{1}{3}
\end{aligned}$$

Suppose we choose a multi-set K_{ij} of size k from K^* by sampling from the probability distribution $\{p_\kappa\}$ constructed above. Let $Z(x) = \alpha(x, c_i, c_j, K_{ij}) = \sum_{\kappa \in K_{ij}} \alpha(x, c_i, c_j, \kappa)$. Then $Z(x)$ is a random variable such that $Z(x) = Y^{(1)}(x) + \cdots + Y^{(k)}(x)$ where $Y^{(\cdot)}(x)$ are independent and identically distributed random variables that have the same distribution as $Y(x)$ defined above. We say that the multi-set K_{ij} is distinguishing for a point x assigned to c_i if $Z(x) < 0$ and for a point x assigned to c_j if $Z(x) > 0$. K_{ij} is *distinguishing* if it is distinguishing for all the points x that are assigned to either c_i or c_j.

Suppose x is assigned to c_i, then $\mathbf{E}[Z(x)] < -k/3$. Using Lemma 5, we get $\Pr[Z(x) \geq 0] \leq \Pr[Z(x) - \mathbf{E}[Z(x)] > k/3] \leq \exp(-k/18)$. By symmetry, if x is assigned to c_j, $\Pr[Z(x) \leq 0] \leq \exp(-k/18)$. For x assigned to either c_i or c_j, let $\mathcal{A}_x$ be the event that K_{ij} is distinguishing for x. Then $\Pr[\overline{\mathcal{A}_x}] \leq \exp(-k/18)$.

Setting $k = 19 \lg m$, $\Pr[\overline{\mathcal{A}_x}] < 1/m$. Since there are only m points, $\Pr[\cup \overline{\mathcal{A}_x}] < 1$ and $\Pr[\cap \mathcal{A}_x] > 0$. With positive probability, the multi-set K_{ij} is distinguishing for the pair c_i, c_j and the assignment of points to centers in the optimal solution. ■

The *separation graph* G^*_{sep} for the optimal solution is an undirected graph on the centers $c_1, \ldots, c_\ell$ in the optimal solution such that the edge (c_i, c_j) is present iff $\text{dist}^{(1)}_{K^*}(c_i, c_j) > 3r$. Consider the run of the algorithm in which the ℓ centers $c_1, c_2, \ldots, c_\ell$ of the optimal solution are chosen and the graph G on the ℓ centers chosen in Step 1 is the separation graph G^*_{sep}; and further for all pairs c_i, c_j such that the edge (c_i, c_j) is present in $G = G^*_{\text{sep}}$ (i.e., $\text{dist}^{(1)}_{K^*}(c_i, c_j) > 3r$), a distinguishing multi-set of dimensions K_{ij} is chosen in Step 2. (The existence of such a multi-set for each pair (c_i, c_j) is guaranteed by Lemma 6.) Call such a run of the algorithm, a *lucky* run.

Lemma 7 *If a point x is assigned to center c_i in the optimal solution, then in a lucky run of the algorithm, c_i is a sink in G_x (and hence the algorithm on its lucky run actually runs and goes past Step 4 to actually return a solution).*

Proof. Consider a center $c_j \neq c_i$. c_i has an edge to or from c_j in G_x iff G^*_{sep} contains the edge (c_i, c_j), i.e., iff $\text{dist}^{(1)}_{K^*}(c_i, c_j) > 3r$. However, if $\text{dist}^{(1)}_{K^*}(c_i, c_j) > 3r$, then a distinguishing multi-set K_{ij} is chosen in the lucky run. By the definition of a distinguishing multi-set, the algorithm directs the edge from c_j to c_i in Step 3. Thus all edges involving c_i are directed towards c_i. Hence c_i is a sink. ■

Lemma 8 *For a lucky run of the algorithm, suppose the optimal solution assigns a point x to center c_i and the algorithm assigns x to c_j then $\text{dist}^{(1)}_{K^*}(c_i, c_j) \leq 3r$.*

Proof. Suppose $c_i \neq c_j$. Then, by Lemma 7, c_i is a sink in G_x. Further, as the algorithm assigns x to c_j, c_j must be a sink in G_x as well. This implies that there is no edge between c_i and c_j in G_x. Hence the edge (c_i, c_j) must be absent in G^*_{sep} which implies that $\text{dist}^{(1)}_{K^*}(c_i, c_j) \leq 3r$. ■

Lemma 9 *In a lucky run of the algorithm, there is a feasible solution to the PIP in Step 5, with objective value at least $|K^*|$.*

Proof. Consider a point x. If x is assigned to c_i in the optimal solution, $\text{dist}^{(1)}_{K^*}(x, c_i) \leq r$. Further, if x is assigned to c_j by the algorithm, then $\text{dist}^{(1)}_{K^*}(c_i, c_j) \leq 3r$. Hence $\text{dist}^{(1)}_{K^*}(x, c_j) \leq 4r$ (by triangle inequality). This implies that for all $\kappa \in K^*$, $\text{dist}_\kappa(x, \sigma(x)) \leq 4r$. Further, this condition holds for all points x. Therefore, all $\kappa \in K^*$ are included in the subset K chosen in Step 5.

We can now construct a feasible solution to the PIP in Step 5 as follows: For all $\kappa \in K^*$, set $y_\kappa = 1$, for $\kappa \notin K^*$, set $y_\kappa = 0$. Then,

$$\begin{aligned} \forall x \quad \sum_{\kappa \in K} y_\kappa \cdot \text{dist}_\kappa(x, \sigma(x)) &= \sum_{\kappa \in K} \text{dist}_\kappa(x, \sigma(x)) \\ &= \text{dist}^{(1)}_{K^*}(x, \sigma(x)) \leq 4r. \end{aligned}$$

Thus the solution is a feasible solution to the PIP. Also, the value of the objective function is $\sum_{\kappa \in K} y_\kappa = \sum_{\kappa \in K^*} 1 = |K^*|$. ■

Applying randomized rounding to solve the PIP [26, 29], we get the following guarantee.

Lemma 10 *In a lucky run of the algorithm, with probability* $1 - poly^{-1}(m)$*, the solution produced by the algorithm has at least* $|K^*|/(1+\epsilon)$ *dimensions and radius* $O(\lg m) \cdot r$*.*

Proof. By Lemma 9, there exists a feasible solution to the PIP of objective value $|K^*|$. Hence, with probability at least $1 - 1/\text{poly}(m)$, randomized rounding returns a solution of objective value $|K^*|/(1+\epsilon)$ which violates the packing constraints by at most a factor of $O(\lg m)$ (See [26, 29]). The violation of the packing constraints implies that the radius of the clustering produced is at most $O(\lg m) \cdot r$. ■

Since the algorithm returns the solution with the most number of dimensions amongst all solutions with radius at most $O(\lg m) \cdot r$, we get the following theorem.

Theorem 11 *With probability* $1 - poly^{-1}(m)$*, the algorithm returns a solution with radius* $O(\lg m) \cdot r$ *and number of dimensions at least* $|K^*|/(1+\epsilon)$*.*

The running time of the algorithm is dominated by the time taken to iterate over subsets of vertices of size ℓ and $O(\lg m)$ size multi-sets of dimensions for every pair of centers; thus the running time is $m^\ell 2^{\binom{\ell}{2}} n^{O(\lg m)} \text{poly}(n)$, which is at most $n^{O(\lg m)}$.

Remarks. (i) The probabilistic construction of distinguishing multi-sets in the proof of Lemma 6 might give the impression that exhaustive search over $O(\lg m)$ sized multi-sets can be replaced by a randomized construction of distinguishing multi-sets, resulting in a polynomial time algorithm for fixed ℓ. Unfortunately, this is not true. The probabilistic construction uses a probability distribution over the dimensions in K^*. Since the algorithm does not know K^*, it cannot perform the randomized multi-set construction.

(ii) Using similar (actually simpler) ideas, one can design an algorithm that runs in time $O(m^{r\ell^2})$ time if the points are on the hypercube and r is the optimal radius. Thus in the case when r, ℓ are both constants, this gives a polynomial time algorithm.

(iii) The approach in Lemma 6 is very similar in spirit to techniques used in [21] in that random sampling is used to 'distinguish' pairs of points.

4 Hardness of hidden cluster problems

This section presents hardness results for the hidden cluster problems we have considered. As it will turn out, the algorithms in Section 3 are close to the best possible, as versions of the hidden cluster problems we consider seem to contain as special cases some notoriously hard problems like CLIQUE, DENSEST SUBGRAPH, etc.

4.1 Hardness with arbitrary number of centers

One of the apparent shortcomings of our algorithms is the runtime has an exponential dependence on the number ℓ of clusters. We prove hardness results and provide evidence that this dependence might be inherent, in that with an unbounded number of centers, the problems are probably very hard to approximate within even very moderate factors.

Consider the following special case: Suppose we have a $m \times n$ 0-1 matrix representing m points in n dimensions and we want to find k dimensions such that in these dimensions we obtain at most ℓ distinct points (this corresponds to having ℓ clusters with radius 0 for both the L_1 and L_∞ norms). (Each of the metrics corresponding to the n dimensions has a particularly simple form: it partitions the m points into two parts, and the distance between two points is 0 if they are in the same side of the partition, and is 1 otherwise.) Here, since the optimal radius is 0, any approximation on the radius is equally good, and the question of interest is how well one can approximate the number of dimensions. We prove that obtaining the exact number of dimensions as the optimum is NP-hard via a reduction from MAX CLIQUE.

Lemma 12 *For the L_1 and L_∞ hidden cluster problems, it is NP-hard to find a solution with the optimum number of dimensions with any finite approximation on the radius.*

Proof. We can reduce CLIQUE to this as follows: Given an instance (G, ℓ) of CLIQUE, the input matrix to our problem is the incidence matrix of G with rows corresponding to vertices and columns to edges in G. Thus each column of G has precisely two 1's. Now, G has a clique of size ℓ iff the matrix has $\binom{\ell}{2}$ columns such that in these columns we have at most $\ell + 1$ distinct patterns. Hence for the hidden cluster problems with $(\ell + 1)$ clusters each of radius 0, it is NP-hard to find the optimum number of dimensions. ■

When we relax the requirement on the dimensions, the reduction used in the proof of the above lemma does not yield anything because, given a graph which has an ℓ-clique it is possible to find a subgraph of size ℓ containing $(1-\epsilon)\binom{\ell}{2}$ edges in quasi-polynomial time [14]. We can give a similar reduction from DENSEST SUBGRAPH, however, and this gives the following:

Lemma 13 *If* DENSEST SUBGRAPH *is $f(N)$ hard to approximate on N-vertex graphs, then for both the L_1 and L_∞ hidden cluster problems with n dimensions, it is hard to approximate the number of dimensions within a factor of $f(\sqrt{n})$ for any finite approximation on the radius.*

Proof. An instance (G, ℓ) of DENSEST SUBGRAPH requires us to find a subgraph of G of size ℓ with maximum number of induced edges. Assume that G is connected and has N vertices. Using the incidence matrix of G as an input to the hidden cluster problem (so that the number of dimensions $n = O(N^2)$ and the number of points equals N), G has a subgraph of size ℓ containing k edges iff the matrix has k columns which have at most $\ell + 1$ distinct patterns. (The proof of this is straightforward, except that in order to show the 'if' part, we need to use the fact that G is connected. The solution to the dimension reduction problem might give rise to isolated edges. Both end points of an isolated edge count as one pattern, but they count as two vertices in translating to a solution of DENSEST SUBGRAPH. We can replace such isolated edges by edges connected to existing components without loss of generality.) This implies that for the hidden cluster problem with $(\ell + 1)$ clusters each of radius 0, it is as hard to approximate the number of dimensions as it is to approximate DENSEST SUBGRAPH. ■

In light of the generally believed hardness of DENSEST SUBGRAPH, the above shows that in order to get a constant factor approximation for the number of dimensions, an exponential dependence of the runtime on ℓ seems essential.

4.2 Hardness with a fixed number of centers

The conventional clustering problems (with just one dimension) are trivially solvable in the case when the number of centers is a constant; this, however, is not the case for the hidden clusters problem. We prove that the L_∞ problem is very hard with just two centers and the L_1 version is in fact hard to approximate even when there is only one center. These results also shed light on why our algorithms L_∞ hidden cluster and L_1 hidden cluster have essentially the best possible performance.

Theorem 14 *For any $\delta > 0$, for the L_∞ hidden cluster problem with n dimensions and ℓ clusters, it is NP-hard (under randomized reductions) to get a $(2-\delta, n^{1-\delta})$-approximation for the radius and number of dimensions respectively, even if $\ell = 2$. In case the centers of the ℓ clusters are specified, then it is in fact hard to get a $(3-\delta, n^{1-\delta})$-approximation.*

Proof. The reduction is from MAX-CLIQUE, and we will prove that the number of dimensions in the L_∞ hidden cluster is as hard to approximate as MAX-CLIQUE even with a slack of $(2-\delta)$ on the cluster radius. Our result will then follow using the inapproximability of MAX-CLIQUE [15]. We describe the reduction for $\ell = 2$ clusters. For $\ell > 2$, we simply add $\ell - 2$ additional points such that they are very far away from all the other points in all dimensions.

Given a graph G, we construct an instance of L_∞ hidden cluster as follows. The points in the hidden cluster problem consist of one point x_v for each vertex v of G and two additional points C and $\bar{C}$. We have one dimension κ_v corresponding to each vertex v of G. Each dimension is just a line; in fact the only coordinates we need to use are $0, 1, 2, 3, 4$ (the distance between two points in any dimension is simply the absolute value of the difference between their coordinates in that dimension). The coordinates of points in dimension κ_v are as follows: x_v has coordinate 0. For all u that are adjacent to v in G, x_u has coordinate 2. For all u that are not adjacent to v in G, x_u has coordinate 4. C has coordinate 1 and $\bar{C}$ has coordinate 3. Note that C has coordinate 1 in all dimensions and $\bar{C}$ has coordinate 3 in all dimensions.

Suppose G has a clique of size k. Then for the constructed instance of L_∞ hidden cluster, we can choose k dimensions and two clusters of radius 1 with centers C and $\bar{C}$. Consider any clique in G of size k. We choose dimension κ_v for every vertex v in the clique. For all vertices v in the clique, x_v is assigned to C; the remaining x_v are assigned to $\bar{C}$. In all the chosen dimensions, for v in the clique, x_v has coordinate either 0 or 2; for v not in the clique, x_v has coordinate either 2 or 4. Thus all points are at distance at most 1 from their assigned center.

We now show that solutions to the constructed L_∞ hidden cluster problem with k dimensions with a small slack in the radius allow us to construct cliques of size k in G.

First, we consider the problem with specified centers. Suppose C and $\bar{C}$ are specified as centers. We know that if G has a clique of size k, there is a solution to the hidden cluster problem with k dimensions and radius 1. Suppose we have an algorithm that returns a clustering with radius $3 - \delta$ times the optimal and uses k dimensions. Every point assigned to C must be at most a distance 2 away from it in all dimensions, hence must have a coordinate $\in [0, 3]$ in the chosen dimensions. Similarly, every point assigned to $\bar{C}$ must have a coordinate $\in [1, 4]$ in the chosen dimensions. If κ_v is one of the chosen dimensions, then x_v must be assigned to C as it is too far away from $\bar{C}$ in dimension κ_v. Now, if κ_u and κ_v are two chosen dimensions, then both x_u and x_v must be assigned to C. Consider dimension κ_u. x_v can have coordinate either 2 or 4. Since it is assigned to C, it must have coordinate 2. But this means that u is adjacent to v in G. Let $X = \{v | x_v \text{ is a chosen dimension}\}$. $|X| = k$. The above discussion implies that X is a clique in G. Thus the problem of maximizing the number of chosen dimensions for L_∞ hidden cluster with specified centers and a slack of $3 - \delta$ on the radius is as hard to approximate as MAX-CLIQUE.

Now consider the problem of finding the hidden clustering where the centers are not specified. Suppose we have an algorithm that returns a clustering with radius $2-\delta$ times the optimal and uses $k > 1$ dimensions. Then every point must be at distance at most 1 from its assigned center (since the instance uses only integer distances). We claim that C must be chosen as a center. If not, let κ_v be a chosen dimension. Since C is not a center, x_v must be a center, as that is the only way to cover x_v within distance 1. But then the only way to cover the rest of the points within distance 1 is to choose $\bar{C}$ as the other center. (Here we use the assumption that the graph G does not have isolated vertices). Thus the two centers must be x_v and $\bar{C}$. But this must hold for each of the $k > 1$ chosen dimensions κ_v which is impossible. This proves that C must be chosen as a center. Now, for each chosen dimension κ_v, x_v must be assigned to C. (If not, then x_v must be chosen as a center and then it is impossible to cover all the points within distance 1). Using the same reasoning as before, we can conclude that if κ_u and κ_v are chosen dimensions then u and v are adjacent in G. Again, as before, let $X = \{v | x_v \text{ is a chosen dimension}\}$. $|X| = k$. The set of vertices X is a clique in G. This implies that the problem of maximizing the number of dimensions for L_∞ hidden cluster with a slack of $2 - \delta$ on the radius is as hard to approximate as MAX-CLIQUE. ■

Since L_∞ hidden cluster is a $(3, 1)$-approximation algorithm, it is close to being the best possible. Of course the algorithm has the same performance even if the centers are specified (and cannot be picked arbitrarily), and hence the above hardness result implies that for the case when the centers are specified, the algorithm is in fact optimal.

We now turn to the L_1-version of the problem: The following theorem shows the NP-hardness of designing any constant factor approximation algorithm for the radius in the L_1 hidden cluster problem, and hence the $O(\lg n)$ factor achieved by our algorithm L_1 hidden cluster is not too far from optimal. The theorem follows from arguments similar to the results of Chekuri and Khanna [7] on the hardness of approximating PIPs.

Theorem 15 *For any $\delta > 0$ and any constant $c > 1$, for the L_1 hidden cluster problem with n dimensions and ℓ centers, it is NP-hard under randomized reductions to find a $(c, n^{1-\delta})$-approximation algorithm for the radius and number of dimensions respectively, even for the case when there is only one center, i.e., $\ell = 1$.*

Proof. The reduction is from the INDEPENDENT SET problem. We use the hardness result of Håstad [15] which states that, for any $\epsilon_0 > 0$, given an n-vertex graph G, it is NP-hard, under randomized reductions, to distinguish between the cases when $\alpha(G) \geq n^{1-\epsilon_0}$ and when $\alpha(G) \leq n^{\epsilon_0}$. Given a graph $G(V, E)$ on n vertices and m edges, we create an instance of L_1 hidden cluster over n dimensions (one for each vertex of G), one center and a radius of 1; the points in fact all lie in the n-dimensional hypercube. We use $\epsilon_0 = \delta/(2\lfloor c \rfloor + 1)$ in the hardness result for INDEPENDENT SET above. The hidden cluster instance has one special point $P_0 = 0^n$, and for each clique C in G of size at most $\lfloor c \rfloor + 1$, there is a point P_C with 1's in the dimensions corresponding to vertices in C and 0's in all other dimensions. Note that this instance thus has $O(n^{\lfloor c \rfloor + 1})$ n-dimensional points.

If G has an independent set I of size α, then pick as center the point P_0 and the dimensions which correspond to the vertices in I. Since I is independent, all points P_e have at most one 1 in the set of dimensions corresponding to I; thus if $\alpha(G) \geq n^{1-\epsilon_0}$, this gives a feasible solution to the L_1 hidden cluster instance with radius 1 and number of dimensions at least $n^{1-\epsilon_0}$.

Conversely, suppose that we have a solution to the L_1 hidden cluster problem with one center that picks k dimensions and achieves a radius of at most c (and hence a radius of at most $c = \lfloor c \rfloor$ as all distances are integral). In case the center is not the point P_0, we can throw away at most $(c+1)$ dimensions (corresponding to dimensions where the center has 1's) and have a feasible solution with center P_0, a radius c and a set K of at least $(k - c - 1)$ dimensions. Consider the set S of vertices corresponding to the dimensions in K; since every point has at most c 1's in the dimensions K, S is a subset of V with no $(c+1)$-clique. By simple Ramsey type arguments (see for instance a similar reduction for proving the hardness of approximating PIPs in [7]), this implies that G (in fact the subgraph of G induced by S) has an independent set $I \subseteq S$ of size at least $|S|^{1/c} \geq (k - c - 1)^{1/c}$. Hence if $k \geq n^{2\epsilon_0 c}$, then $\alpha(G) > n^{\epsilon_0}$, or equivalently, if $\alpha(G) \leq n^{\epsilon_0}$ then the maximum number of dimensions for a cluster of radius c in the L_1 hidden cluster instance is at most $n^{2\epsilon_0 c}$.

Combining the above two arguments, we conclude that it is NP-hard (under randomized reductions) to approximate the number of dimensions within a factor of $n^{1-\epsilon_0-2\epsilon_0 c} = n^{1-\delta}$ even with a factor c slack in the cluster radius. ∎

5 Dimension reduction problems

5.1 Entropy maximization

The problem is to pick K so that the entropy of the random variable $U(S_{|K})$ is maximized. We consider two kinds of problems: (i) FIXED-DIMENSION-MAX-ENTROPY: Given k, find K such that $|K| = k$ so as to maximize the entropy $H(U(S_{|K}))$, and its dual (ii) FIXED-ENTROPY-MIN-DIMENSION: Given E, find K such that $H(U(S_{|K})) \geq E$ and $|K|$ is minimized.

We can use the natural greedy algorithm to solve the two problems. The idea is to repeatedly add a dimension j to the currently picked set K of dimensions; j is chosen so as to maximize the entropy of $U(S_{|K \cup \{j\}})$. We repeat this step until we pick k dimensions or achieve the specified entropy bound E. The analysis of the algorithm uses the *subadditivity* of the entropy function, and follows the analysis of the greedy algorithm for the SET COVER and MAX COVERAGE problems.

Lemma 16 *Suppose K_1 and K_2 are two subsets of dimensions. Let $e_1 = H(U(S_{|K_1}))$ and $e_2 = H(U(S_{|K_2}))$. If $e_1 > e_2$, then there exists $j \in K_1$ such that the $H(U(S_{|K_2 \cup \{j\}})) \geq e_2 + (e_1 - e_2)/|K_1|$.*

Proof. For dimension j, let Y_j be the random variable $U(S_{|j})$ and for a subset K of dimensions, let Y_K be the random variable $U(S_{|K})$. Then for a subset K of dimensions, the entropy of $U(S_{|K})$ equals $H(Y_K)$.

$$\begin{aligned} e_1 - e_2 &= H(Y_{K_1}) - H(Y_{K_2}) \\ &\leq H(Y_{K_1 \cup K_2}) - H(Y_{K_2}) \\ &= H(Y_{K_1 \cup K_2} | Y_{K_2}) \\ &\leq \sum_{j \in K_1 \cup K_2} H(Y_j | Y_{K_2}) \\ &= \sum_{j \in K_1} H(Y_j | Y_{K_2}), \end{aligned}$$

where the inequalities are a consequence of subadditivity. Choose $j^* \in K_1$ that maximizes $H(Y_{j^*} | Y_{K_2})$. Then $H(Y_{j^*} | Y_{K_2}) \geq (e_1 - e_2)/|K_1|$, and hence

$$H(U(S_{|K_2 \cup \{j\}})) = H(Y_{K_2 \cup \{j^*\}}) \geq \\ H(Y_{K_2}) + H(Y_{j^*} | Y_{K_2}) \geq e_2 + \tfrac{e_1 - e_2}{|K_1|}\ . \ \blacksquare$$

Using Lemma 16 and the analysis of the greedy algorithm for MAX COVERAGE, we get:

Theorem 17 *There is a polynomial time $(1 - e^{-1})$-approximation algorithm for* FIXED-DIMENSION-MAX-ENTROPY.

Theorem 18 *There is a polynomial time $O(\lg m)$-approximation for* FIXED-ENTROPY-MIN-DIMENSION.

Proof. Suppose the optimal solution is a set of k dimensions K^* such that $H(U(S_{|K^*})) = e^*$. Let K_i be the set of dimensions constructed after i steps of the greedy algorithm; $|K_i| = i$. Let $e_i = H(U(S_{|K_i}))$. We can show that $e_{i+1} - e_i > c/m$ for some constant c. Using Lemma 16, we get $(e^* - e_{i+1}) \leq (e^* - e_i)(1 - \frac{1}{|K^*|})$. Since $e^* \leq \lg m$, this implies that in $O(|K^*| \lg m)$ steps, we have $e^* - e_i \leq c/m$. But this means that we reach the target entropy in at most one more step. $\blacksquare$

The problem FIXED-ENTROPY-MIN-DIMENSION with $E = \lg m$ is equivalent to the *distinct vectors* problem. The following theorem follows from the hardness for distinct vectors (Theorem 20).

Theorem 19 *Unless* P=NP*, there exists $c > 0$, such that* FIXED-ENTROPY-MIN-DIMENSION *is hard to approximate to within a factor of $c \lg n$ where n is the number of dimensions.*

5.2 Distinct vectors

We now consider the feature selection problem where the goal is to pick the minimum number of dimensions that can still distinguish the given points. Using a reduction to SET COVER, it is easy to see that this problem can be approximated within a factor of $O(\lg n)$ (if n is the number of dimensions; the number of points is bounded by a polynomial in n). The special structure of this problem might raise hopes that one can in fact do much better, but, we show that that this is not the case. This result is very similar to a folklore result that it is as hard as set-cover to find the smallest set of features needed to distinguish a specified vector from all the others; our proof also follows from a reduction from SET COVER.

Theorem 20 *Unless* P=NP*, there is a constant $c > 0$ such that the distinct vectors feature selection problem is hard to approximate within a factor of $c \lg n$ where n is the number of dimensions.*

Proof. The reduction is from the SET COVER problem. We start from a hard instance $\mathcal{SC} = (\mathcal{F}, U, k)$ of SET COVER with a family $\mathcal{F}$ of n sets $S_1, S_2, \ldots, S_n$ over an m-element universe $U = \{u_1, u_2, \ldots, u_m\}$ (we assume that $m = n$ but retain use of both symbols for clarity of presentation, we also assume that m is a power of two, and that $k \geq \sqrt{n}$) such that it is NP-hard to distinguish between the "YES" case when there is a set cover of size k and the "NO" case when every set cover has size greater than $c_0 k \lg m$ (here $c_0 > 0$ is an absolute constant).

We will create an instance $\mathcal{I}$ of the distinct vectors problem with $M = 2m+1$ points over $N = n+\lg m$ dimensions, which we shall describe as an $M \times N$ 0-1 matrix V. The leftmost $m \times n$ sub-matrix of V is simply the element-set incidence matrix of the set cover instance $\mathcal{SC}$. Thus the first m rows of V correspond to the points $U_1, U_2, \ldots, U_m$ associated with the elements $u_1, u_2, \ldots, u_m$ of U; the next m rows will correspond to $U'_1, U'_2, \ldots, U'_m$ where each U'_i will be identical to U_i in the last $\lg m$ dimensions and will be all 0's in the first m dimensions. This ensures that in order to distinguish U_i from U'_i, we must pick a dimension corresponding to a set S_j such that $u_i \in S_j$. Hence any solution to the distinct vectors problem, when projected to the first m dimensions, must pick a set cover for the instance $\mathcal{SC}$. Thus, if we start from a NO instance of set cover, we need at least $c_0 k \lg m$ dimensions in any feasible solution to the instance $\mathcal{I}$.

Now for the case when the instance $\mathcal{SC}$ is a YES instance, so that there is a set cover $\{S_j : j \in \mathcal{C}\}$ with $|\mathcal{C}| \leq k$. Picking a set cover alone, however, does not distinguish all pairs of points, and the role of the last $\lg m$ dimensions will be to ensure this at a low cost. The coordinates of the points U_i and U'_i in the last $\lg m$ dimensions will simply be the $\lg m$-bit binary representation of $(i-1)$. Consider the following solution to instance I: Pick the dimensions corresponding to the set cover $\mathcal{C}$ together with the last $\lg m$ dimensions. We now argue that this is a feasible solution to the instance $\mathcal{I}$. Since for $i \neq j$, the binary representation of i and j differ in at least one coordinate, picking the last $\lg m$ columns of the instance $\mathcal{I}$ ensures that all pairs (U_i, U_j), (U'_i, U'_j) and (U_i, U'_j) are distinguished for every $i \neq j$. Also since $\mathcal{C}$ is a set cover, for $1 \leq i \leq m$, the pair (U_i, U'_i) is distinguished by some dimension in $\mathcal{C}$. Hence the above is indeed a feasible solution to the instance $\mathcal{I}$ that picks $|\mathcal{C}| + \lg m \leq k + \lg m$ dimensions.

Hence the distinct vectors feature selection problem is NP-hard to approximate within a factor

$$\frac{c_0 k \lg m}{k + \lg m} \geq \frac{c_0 \lg n}{1 + \lg n / \sqrt{n}} \geq \frac{c_0}{2} \lg n.$$

■

We now consider some generalizations of the distinct vectors problem. Given a 0-1 matrix, consider the problem of choosing k dimensions so as to maximize the number of distinct rows. We call this the MAX-DISTINCT-POINTS problem. The dual problem of minimizing the number of dimensions so that there are ℓ distinct rows is called the MIN-ℓ-DISTINCT-DIMENSION problem.

We give reductions from DENSEST SUBGRAPH to both these problems which show that good approximations for either of them would imply good approximations for DENSEST SUBGRAPH. The input to MAX-DISTINCT-POINTS and MIN-ℓ-DISTINCT-DIMENSION in both our reductions is a matrix M which is the incidence matrix of $G = \langle V, E \rangle$ (with rows corresponding to E and columns corresponding to V) together with an additional $|V| + 1$ rows—one row corresponding to each vertex v with a 1 in the column corresponding to v and 0 elsewhere and the last row with all zeros. We will use the following property connecting M and G: For every proper induced subgraph of G with k vertices and m edges, the corresponding k columns in M have $m + k + 1$ distinct rows and vice-versa (using the fact that G is connected and the subgraph is proper).

Theorem 21 *An α-approximation algorithm for* MAX-DISTINCT-POINTS *implies a 2α-approximation for* DENSEST SUBGRAPH.

Proof. Let (G, k) be an instance of DENSEST SUBGRAPH. Suppose the optimal solution to DENSEST SUBGRAPH on G is a subgraph with k vertices and m edges. We use the incidence matrix M as input to

MAX-DISTINCT-POINTS, requiring that we select k columns so as to maximize the number of distinct rows. The optimal solution is $m + k + 1$.

Suppose the α-approximation algorithm for MAX-DISTINCT-POINTS returns a solution with k columns and $m'+k+1$ rows. This corresponds to a subgraph of G with k vertices and m' edges such that $m'+k+1 \geq (m + k + 1)/\alpha$ implying that $m' \geq m/\alpha - (k + 1)(1 - 1/\alpha)$. Now, we can also get a solution that has at least k edges. The two solutions together have at least m/α edges, implying that the better of the two has at least $m/2\alpha$ edges. ■

Theorem 22 *An α-approximation algorithm for* MIN-ℓ-DISTINCT-DIMENSION *implies a $\alpha(\alpha+1)$-approximation for* DENSEST SUBGRAPH.

Proof. Let (G, k) be an instance of DENSEST SUBGRAPH.

Suppose the optimal solution to DENSEST SUBGRAPH on G is a subgraph with k vertices and m edges. We use the incidence matrix M as input to MIN-ℓ-DISTINCT-DIMENSION, requiring that we find $m+k+1$ distinct rows with the smallest number of columns. The optimal solution is k. (If less than k columns give more than $m + k + 1$ distinct rows, this contradicts the optimality of the densest subgraph solution).

Suppose the α-approximation algorithm for MIN-ℓ-DISTINCT-DIMENSION returns a solution with k' columns and $m' + k' + 1$ rows. This corresponds to a subgraph of G with k' vertices and m' edges such that $k' \leq \alpha k$ and $m' + k' + 1 \geq m + k + 1$ implying that $m' \geq m - k'$.

For DENSEST SUBGRAPH, we wish to obtain a k vertex subgraph. Using random sampling (and derandomizing), we can get such a subgraph with $m'(k/k')^2$ edges. This has at least $m/\alpha^2 - k/\alpha$ edges. Also, we can get a subgraph with k edges. The better of the two contains $m/(\alpha(\alpha + 1))$ edges. ■

6 Further work

There are a number of interesting problems in the feature selection framework. We mention a few:

(i) **Stopword elimination**: Given a clustering $C_1, \ldots, C_\ell$ and α, maximize K such that for each pair of clusters, C and C', $\text{dist}_{|K}(C, C') \geq \alpha k$. This is tantamount to eliminating stopwords to make latent document clusters apparent.

(ii) **Metric embedding**: Given that the Hamming distance between any pair of vectors in S is at least αn for some $\alpha \in (0, 1)$ and a $\beta < \alpha$, minimize $|K|$ such that such that the Hamming distance between any pair of vectors in $S_{|K}$ is at least $\beta|K|$. This problem corresponds to asking whether a code with distance α contains a sub-code with distance at least β.

(iii) **Min dimension perceptron**: given $S = R \cup B$ and a guarantee that there is a hyperplane H separating R from B, find the smallest K such that there is a hyperplane separating $R_{|K}$ and $B_{|K}$.

It will be interesting to study these and other feature selection problems from the algorithms and complexity point of view.

References

[1] C. Aggarwal, C. Procopiuc, J. Wolf, P. Wu, and J. S. Park. Fast algorithms for projected clustering. *Proc. SIGMOD*, pp. 61–72, 1999.

[2] R. Agrawal, J. Gehrke, D. Gunopulos, and P. Raghavan. Automatic subspace clustering of high dimensional data for data mining applications. *Proc. SIGMOD*, pp. 94–105, 1998.

[3] Y. Aumann and Y. Rabani. An $O(\lg k)$ approximate min-cut max-flow theorem and approximation algorithm. *SIAM J. Comput.*, 27(1):291–301, 1998.

[4] J. Bourgain. On Lipschitz embedding of finite metric spaces in Hilbert space. *Israel J. Math.*, 52:46–52, 1985.

[5] S. Chakrabarti, B. Dom, R. Agrawal, and P. Raghavan. Using taxonomy, discriminants, and signatures to navigate in text databases. *Proc. 23rd VLDB*, 1997.

[6] S. Chakrabarti, B. Dom, and P. Indyk. Enhanced hypertext categorization using hyperlinks. *Proc. SIGMOD*, 1998.

[7] C. Chekuri and S. Khanna. On multidimensional packing problems. *Proc. 10th SODA*, pp. 185–194, 1999.

[8] M. Dash and H. Liu. Handling large unsupervised data via dimensionality reduction. *Proc. SIGMOD Workshop on research issues in data mining and knowledge discovery*, 1999.

[9] S. Dasgupta. Learning mixtures of Gaussians. *Proc. 40th FOCS*, 1999. To appear.

[10] S. Deerwester, S. Dumais, G. Furnas, T. Landauer, and R. Harshman. Indexing by latent semantic analysis. *J. Amer. Soc. for Inf. Sci.*. 41(6):391–407, 1990.

[11] I. S. Dhillon and D. S. Modha. Concept decompositions for large sparse text data using clustering. Manuscript, 1999.

[12] M. E. Dyer and A. M. Frieze. A simple heuristic for the p-center problem. *Oper. Res. Lett.*, 3:285–288, 1985.

[13] U. M. Fayyad, G. Piatetsky-Shapiro, P. Smyth, and R. Uthurusamy, editors. *Advances in Knowledge Discovery and Data Mining*. AAAI/MIT Press, 1996.

[14] U. Feige and M. Seltser. On the densest k-subgraph problems. *CS TR 97-16*, Weizmann Institute of Science, 1997.

[15] J. Håstad. Clique is hard to approximate within $n^{1-\epsilon}$. *Proc. 37th FOCS*, pp. 627–636, 1996.

[16] D. S. Hochbaum and D. B. Shmoys. A best possible approximation algorithm for the k-center problem. *Math. Oper. Res.*, 10:180–184, 1985.

[17] G. John, R. Kohavi, and K. Pfleger. Irrelevant features and the subset selection problem. Proc. ML, 1994.

[18] W. B. Johnson and J. Lindenstrauss. Extensions of Lipschitz mappings into a Hilbert space. *Contemporary Math.*, 26:189–206, 1984.

[19] D. Koller and M. Sahami. Towards optimal feature selection. *Proc. ICML*, pp. 284–292, 1996.

[20] D. Koller and M. Sahami. Hierarchically classifying documents using very few words. *Proc. 14th ML*, pp. 170–178, 1997.

[21] E. Kushilevitz, R. Ostrovsky, and Y. Rabani. Efficient search for approximate nearest neighbor in high dimensional spaces. *Proc. 30th STOC*, pp. 614–623, 1998.

[22] N. Linial, E. London, and Y. Rabinovich. The geometry of graphs and some of its algorithmic applications. *Proc .35th FOCS*, 577-591, 1994.

[23] R. Motwani and P. Raghavan. *Randomized Algorithms*. Cambridge University Press, 1995.

[24] C. Papadimitriou, P. Raghavan, H. Tamaki, and S. Vempala. Latent semantic indexing: A probabilistic analysis. *Proc. PODS*, 1997.

[25] C. Papadimitriou and M. Yannakakis. On Limited Non-determinism and the complexity of the VC-dimension. *IEEE Conf. on Comp. Complexity*, 1993, pp 12-18.

[26] P. Raghavan and C. D. Thompson. Randomized rounding: A technique for provably good algorithms and algorithmic proofs. *Combinatorica*, 7:365-374, 1987.

[27] G. Salton. *Automatic Text Processing*. Addison Wesley, 1989.

[28] L. Schulman. Clustering for edge-cost minimization. Manuscript, 1999.

[29] A. Srinivasan. Improved approximations of packing and covering problems. In *Proc. 27th STOC*, pp 268-276, 1995.

[30] G.K. Zipf. Human behaviour and the principle of least effort. Hafner, New York, 1949.