Scalable Exemplar Clustering and Facility Location via Augmented Block Coordinate Descent with Column Generation

Ian E.H. Yen
Computer Science Department
University of Texas at Austin

Dmitry Malioutov
IBM Research

Abhishek Kumar
IBM Research

Abstract

In recent years exemplar clustering has become a popular tool for applications in document and video summarization, active learning, and clustering with general similarity, where cluster centroids are required to be a subset of the data samples rather than their linear combinations. The problem is also well-known as facility location in the operations research literature. While the problem has well-developed convex relaxation with approximation and recovery guarantees, its number of variables grows quadratically with the number of samples. Therefore, state-of-the-art methods can hardly handle more than 10^4 samples (i.e. 10^8 variables). In this work, we propose an Augmented-Lagrangian with Block Coordinate Descent (AL-BCD) algorithm that utilizes problem structure to obtain closed-form solution for each block sub-problem, and exploits low-rank representation of the dissimilarity matrix to search active columns without computing the entire matrix. Experiments show our approach to be orders of magnitude faster than existing approaches and can handle problems of up to 10^6 samples. We also demonstrate successful applications of the algorithm on world-scale facility location, document summarization and active learning.

1 Introduction

Exemplar clustering is a popular unsupervised learning approach, which, unlike traditional clustering, guarantees the cluster centroids to be samples from the data set [15, 25, 35]. This requirement is important in a variety of applications where it may not make sense to take linear combinations of the data-points, such as images, news articles, or chemical compounds. Furthermore, it also applies to situations where the set of objects is characterized by a set of pairwise non-Euclidean distances (for example string edit-distances, or transportation distances) and hence computing averages, a central step in algorithms such as k-means, is not well defined. Furthermore, exemplar clustering also allows one to specify a weight or a cost associated with each exemplar so as to balance the individual costs of the exemplars versus the quality of the clustering. Some of the applications where such weights are required include active learning for image and document classification [7], document summarization [14], and facility location in the Operations Research literature [3]. For example, in the context of active learning the exemplars represent a new subset of data-points to be labeled, so they are chosen to balance the expected information gain versus their coverage of the other unlabeled examples.

A direct formulation of exemplar clustering leads to the combinatorial k-medoid problem, which is NP-Hard. Thus different approximation schemes have been proposed for the problem. Partitioning Around Medoids (PAM) is a popular local-search method [13], which however typically requires a large number of random restarts to provide competitive solutions. A message-passing scheme based on a variant of loopy belief propagation has also been proposed to solve the problem [8] which however does no guarantee convergence. The maximization version of the problem can be cast as a submodular maximization problem, where a greedy algorithm can achieve $1 - 1/e$ approximation guarantee [2, 23], while the current tightest approximation guarantees for the problem are achieved by the convex (Linear Program) relaxation approach [1, 18, 19], which can even find exact solutions if the data satisfy certain clustering condition [25, 35].

Appearing in Proceedings of the 19^{th} International Conference on Artificial Intelligence and Statistics (AISTATS) 2016, Cadiz, Spain. JMLR: W&CP volume 51. Copyright 2016 by the authors.

A significant challenge in scaling the existing exemplar clustering approaches to larger data-sets has been in the need to specify a pairwise distance matrix, which already requires $O(N^2)$ storage with N data-points. Current state-of-the-art approaches to exemplar clustering and facility location can hardly scale beyond data-sets of size $10,000$, especially for the convex approach, where a Linear Program with N^2 variables is extremely hard to solve [3, 9, 35].

In this paper, we propose a new convex optimization algorithm for the convex relaxation approach based on Augmented-Lagrangian method with column-wise block coordinate descent (AL-BCD) that, by utilizing the column-separable structure of group norm and the row-separable structure of simplex constraint, obtains closed-form solution for each column sub-problem. Furthermore, we develop a greedy column generation procedure based on the low-rank (or sparse) decomposition of the dissimilarity matrices, which can find active columns without computing the entire matrix and thus reduce the complexity per iteration from quadratic to linear w.r.t. the number of samples. Experiments show our approach to be orders of magnitude faster than existing approaches and can handle problems with more than 10^5 samples. We demonstrate our approach on a World-scale facility location problem with over 1 million nodes, an active learning experiment in document classification, and apply the technique for document summarization.

2 Problem Setup

Exemplar clustering aims to find a small number of representatives that summarizes the entire data set in the sense that each sample has high similarity to the representative of its cluster. In a k-medoid formulation, given a data set $\{\boldsymbol{x}_i\}_{i=1}^N$, the problem can be expressed as the following optimization problem

$$\min_{z_i \in \{1,..,K\}, \mu_k \in \mathcal{E}} \quad \sum_{i=1}^{N} D(\boldsymbol{x}_i, \boldsymbol{\mu}_{z_i}), \tag{1}$$

where $D(.,.)$ is a dissimilarity function and $\mathcal{E}$ denotes the set of candidate representatives, which usually is the same as the set of data points $\mathcal{E} = \{\boldsymbol{\mu}\}_{j=1}^M$, but can be also explicitly given in cases such as facility location. The problem (1) has convex relaxation with well-developed clustering-recovery guarantees [6, 25, 35] and rounding guarantees [18, 19]. The relaxation is defined on an assignment matrix $W \in [0,1]^{N,M}$, where W_{ij} denotes whether i-th data sample is assigned to the exemplar j, and a cluster is implied by the samples assigned the same exemplar. Therefore, any non-zero columns of W will imply a cluster and the ℓ_∞-group norm $\|W\|_{\infty,1} = \sum_{j\in[M]} \max_{i\in[N]} |W_{ij}|$ is a convex relaxation of the number of non-zeros columns. The Exemplar Clustering problem then can be expressed as

$$\begin{aligned} \min_{W\in[0,1]^{N\times M}} \quad & F(W) = tr(\mathbb{D}^T W) \\ s.t. \quad & W\mathbf{1}_M = \mathbf{1}_N,\ \|W\|_{\infty,1} \le K. \end{aligned} \tag{2}$$

where $\mathbb{D}$ denotes $N \times M$ dissimilarity matrix, and $\mathbf{1}_n$ denotes an $n \times 1$ vector of all ones. In case the number of clusters K is not known a-priori, one can solve another formulation

$$\begin{aligned} \min_{W\in[0,1]^{N\times M}} \quad & F(W) = tr(\mathbb{D}^T W) + \lambda\|W\|_{\infty,1} \\ s.t. \quad & W\mathbf{1}_M = \mathbf{1}_N, \end{aligned} \tag{3}$$

which is equivalent to (2) in the sense that for any K there is a $\lambda(K)$ s.t. (3) and (2) have the same optimal solutions. The convex relaxation (3), as shown in [25, 35], exactly recover the solution of (1) if there exists clustering with good separation. Furthermore, (3) is of the same form to the facility location problem [3], where one can employ a *rounding procedure* to find approximate solution with 1.488 approximation ratio to the optimal integer solution [18]. In this paper, we will focus on formulation (3) to address both large-scale exemplar clustering and facility location.

However, in current practice, algorithms for exemplar clustering require construction and repeated access to the $N \times M$ dissimilarity matrix $\mathbb{D}$, which, for $N, M > 10^5$, may not even fit into memory, and requires $O(MNd)$ construction time, where d is the time required to compute one pair of dissimilarity.

In the next section, we show how to overcome this issue via a low-rank or sparse decomposition of $\mathbb{D}$, and in section 4, we propose an algorithm that can explicitly utilize such decomposition to avoid computing matrix $\mathbb{D}$ and thus avoid the quadratic complexity w.r.t. number of samples.

3 Exact & Approximate Dissimilarity Matrix Decomposition

In many applications, the dissimilarity matrix $\mathbb{D}$ has a tractable representation, for example it can be exactly or approximately factorized into a product of low-rank or sparse matrices. We now discuss some special cases.

3.1 Exact Dissimilarity Decomposition for Bregman Divergence

Bregman Divergence is a family of dissimilarity measures that includes several popular measures such as square Euclidean Distance, KL-divergence, Square Mahalanobis distance, and Itakura-Saito distance.

Given a function $f(x) : \mathbb{R}^d \rightarrow \mathbb{R}$, the Bregman Divergence between sample $\boldsymbol{x}_i$ and an exemplar $\boldsymbol{\mu}_j$ is defined as

$$D(\boldsymbol{x}_i, \boldsymbol{\mu}_j) = f(\boldsymbol{x}_i) - f(\boldsymbol{\mu}_j) - \langle \nabla f(\boldsymbol{\mu}_j), \boldsymbol{x}_i - \boldsymbol{\mu}_j \rangle, \quad (4)$$

where the first term is related only to i, the second term is related only to j, and the only term involving both i and j is the d-dimensional inner product $-\langle \nabla f(\boldsymbol{\mu}_j), \boldsymbol{x}_i \rangle$. Therefore, let X be the $N \times d$ data matrix with $X_{i,:} = \boldsymbol{x}_i^T$. The dissimilarity matrix can be decomposed as

$$\mathbb{D} = \boldsymbol{f}_x \boldsymbol{1}_M^T + \boldsymbol{1}_N \boldsymbol{f}_\mu^T - X F^T. \quad (5)$$

where $\boldsymbol{f}_x$ is N by 1 with $\boldsymbol{f}_{xi} = f(x_i)$, $\boldsymbol{f}_\mu$ is M by 1 with $\boldsymbol{f}_{\mu i} = \langle \nabla f(\mu_j), \mu_j \rangle - f(\mu_i)$, and F is M by d matrix with j-th row being $\nabla f(\mu_j)^T$. Therefore, the dissimilarity matrix formed by Bregman Divergence between d-dimensional objects has decomposition (5) with rank $d+2$. The matrix-vector product operation on the dissimilarity matrix can be thus computed in $O(Nd+Md)$ time, which is much smaller than $O(MN)$. Note in some applications such as Natural Language Processing, dimension d can be quite large, but data matrix X and F are very sparse, in which case the time for matrix-vector product can be bounded by $O(nnz(X) + nnz(F)) \ll O(Md + Nd)$.

3.2 Approximate Dissimilarity Matrix Decomposition

For a general dissimilarity (similarity) measure, there is no exact low-rank representation of $\mathbb{D}$, but we can find an approximate one via general techniques for any dissimilarity or similarity measure such as *Anchor Graph Approximation* [21] or *Nystrom Method* [17], which can be constructed with cost $O(Nmr + r^3)$, where m is number of anchors chosen and r is the rank for approximation. In some cases such as Facility Location, it is known that $\mathbb{D}$ can be approximated well by a low-rank matrix since the Euclidean Distance matrix is computed from a 2-dimensional space, for which we can use Matrix Completion [12, 27] to compute a low-rank representation from sampled pairwise distances.

No matter which approximation method we use, it leads to a decomposition of the form

$$\mathbb{D} = U V^T, \quad (6)$$

where U, V are of size $N \times r$ and $M \times r$. Then one can compute matrix-vector product in time $O(Nr + Mr)$.

The following theorem shows that as long as the approximation is accurate enough, one can bound the sub-optimality of the solution obtained from the low-rank dissimilarity matrix. Proofs are in the appendix.

Theorem 1. *Suppose the low-rank approximation $\hat{\mathbb{D}}$ has $\|\hat{\mathbb{D}} - \mathbb{D}\|_{1,\infty} \leq \epsilon$, where $\|A\|_{1,\infty} = \sum_i \max_j |A_{ij}|$. Then we have*

$$tr(\mathbb{D}^T \hat{W}) \leq tr(\mathbb{D}^T W^*) + 2\epsilon, \quad (7)$$

where $\hat{W}$, W^* are solutions to (2) *with dissimilarity matrix $\hat{\mathbb{D}}$ and $\mathbb{D}$ respectively.*

Theorem 2. *Suppose the low-rank approximation $\hat{\mathbb{D}}$ has $\|\hat{\mathbb{D}} - \mathbb{D}\|_{1,\infty} \leq \epsilon$, where $\|A\|_{1,\infty} = \sum_i \max_j |A_{ij}|$. Then we have*

$$F(\hat{W}) \leq F(W^*) + 2\epsilon, \quad (8)$$

where $F(.)$ denotes the objective function in (3)*, $\hat{W}$, W^* are solutions to* (3) *with dissimilarity matrix $\hat{\mathbb{D}}$ and $\mathbb{D}$ respectively.*

In addition, if the optimal clustering is strong enough, one can recover the optimal solution even using a low-rank approximation.

Corollary 1. *Suppose the optimal solution W^* of* (2) *(or* (3)*) is strong in the sense that*

$$F(W) - F(W^*) \geq \delta$$

for any corner point $W \neq W^*$ of (2)*. Then if $\|\hat{\mathbb{D}} - \mathbb{D}\|_{1,\infty} < \delta$, the solution $\hat{W}$ of* (2) *(or* (3)*) with dissimilarity matrix $\hat{\mathbb{D}}$ has $\hat{W} = W^*$, where W^* is the solution of* (2) *(or* (3)*) with exact matrix $\mathbb{D}$.*

4 Optimization Algorithm

The low-rank (or sparse) representation introduced in section 3, unfortunately, cannot directly reduce the computational cost for most of the existing optimization methods. In particular, existing optimization approaches [3, 6, 8, 16, 35, 36] require access to every entry of the dissimilarity matrix, and thus they still need $O(MNr)$ time to construct the matrix. In this section, we show that by combining Augmented-Lagrangian with Block Coordinate Descent (AL-BCD), one can utilize problem structure to obtain a closed-form solution for each column sub-problem, and exploit the decomposition of dissimilarity matrices to do column generation without computing the entire matrix.

4.1 Augmented Lagrangian with Block Coordinate Descent (AL-BCD)

The convex optimization problem (3) has special structure where the simplex constraints are row-separable while the group norm $\|W\|_{\infty,1}$ is column-separable. Therefore, if we introduce dual variables $\boldsymbol{\alpha} \in \mathbb{R}^N$ for the simplex constraints and form its Augmented Lagrangian

$$\begin{aligned}\mathcal{L}(W;\boldsymbol{\alpha}) =& tr(\mathbb{D}^T W) + \lambda \|W\|_{\infty,1} \\ &+ \boldsymbol{\alpha}^T(W\mathbf{1}-\mathbf{1}) + \frac{\rho}{2}\|W\mathbf{1}-\mathbf{1}\|^2,\end{aligned} \tag{9}$$

with some parameter ρ, the sub-problem (9) will have diagonal Hessian sub-matrix for each column of variables, which means we will have closed-form solution if we minimize (9) w.r.t. only one column $W_{:,j}$. In particular, the Augmented Lagrangian method alternates between minimizing the Augmented Lagrangian (AL)

$$W^{t+1} = \underset{W \geq 0}{argmin}\ \mathcal{L}(W, \boldsymbol{\alpha}^t) \tag{10}$$

and updating the dual variables

$$\boldsymbol{\alpha}^{t+1} = \eta(W\mathbf{1}-\mathbf{1}) + \boldsymbol{\alpha}^t \tag{11}$$

where η is a step size parameter. The AL sub-problem (10) is of the form:

$$\mathcal{L}(W;\boldsymbol{\alpha}) = f(W) + h(W), \tag{12}$$

where $h(W) = \lambda\|W\|_{\infty,1}$ is the non-smooth part, and $f(W)$ is the smooth part containing remaining terms. Since $h(W)$ is column separable, we can solve (12) via a column-wise Block Coordinate Descent, which optimizes a column of W at a time. Minimizing (12) w.r.t. j-th column results in the following subproblem

$$\begin{aligned}\min_{\boldsymbol{d}_j} \quad & h_j(W_{:,j}+\boldsymbol{d}_j) + \nabla_j f(W)^T \boldsymbol{d}_j + \frac{1}{2}\boldsymbol{d}_j^T \nabla^2_{jj} f(W) \boldsymbol{d}_j \\ s.t. \quad & W_{ij} + d_{ij} \geq 0\end{aligned} \tag{13}$$

where $h_j(W_{:,j}) = \|W_{:,j}\|_\infty$,

$$\nabla_j f(W) = \mathbb{D}_{:,j} + \rho \boldsymbol{r} \tag{14}$$

$$\nabla^2_{jj} f(W) = \rho I, \tag{15}$$

and

$$\boldsymbol{r} = W\mathbf{1} - \mathbf{1} + \boldsymbol{\alpha}^t/\rho. \tag{16}$$

Since (13) is minimization of a quadratic function of diagonal Hessian matrix, we can derive its closed-form solution as

$$\boldsymbol{d}_j^* = \mathbf{prox}_{h_j/\rho}\left(\left[W_{:,j} - \frac{\nabla_j f(W)}{\rho}\right]_+\right) - W_{:,j}, \tag{17}$$

where $\mathbf{prox}_{h_j/\rho}(.)$ is the proximal operator of the infinity vector norm $\frac{\lambda}{\rho}\|W_{:,j}\|_\infty$. For a non-negative vector $\boldsymbol{v}$, assuming $\boldsymbol{v}$ is sorted such that $v_1 \geq v_2 ... \geq v_N$, the proximal operation can be computed as

$$W_{ij} + d^*_{ij} = \begin{cases} \frac{1}{m^*}\left[(\sum_{i=1}^{m^*} v_i) - \frac{\lambda}{\rho}\right]_+, & i \leq m^* \\ v_i, & i > m^*. \end{cases} \tag{18}$$

where $m^* = arg\max_m \ \frac{1}{m}\left[(\sum_{i=1}^{m} v_i) - \lambda/\rho\right]$ [20].

Algorithm 1 Randomized AL-BCD

```
1. Initialize $\alpha^0 = 0$, $W^0 = 0$.
for $t = 1, ..., T$ (outer iteration) do
  for $s = 1, ..., M$ do
    2.1.1. Draw $j \in [M]$ uniformly at random.
    2.1.2. Update $j$-th column of $W$ via (17).
  end for
  2.2. Update $\alpha^t$ by (11).
end for
```

Algorithm 2 AL-BCD with Column Generation

```
1. Initialize $\alpha^0 = 0$, $W^0 = 0$, $A^{(t)} = \emptyset$.
for $t = 1, ..., T$ (outer iteration) do
  2.1. Generate a set of greedy columns $S_L$ based on criteria (24).
  2.2. Add $S_L$ to active column set $A^{(t)}$.
  2.3. Solve (10) w.r.t. columns $j \in A^{(t)}$.
  2.4. Remove $\{j \mid W^{(t)}_{:,j} = 0\}$ from $A^{(t)}$.
  2.5. Update $\alpha^t$ by (11).
end for
```

This results in the first version of AL-BCD, summarized in Algorithm 1, which alternates between the update of dual variables and the minimization of AL function (10) via Randomized (columnwise) Block Coordinate Descent. Note that instead of solving the AL subproblem (10) exactly, Algorithm 1 performs only M block minimization steps before each update of dual variables (11), which as we show in section 5, suffices for achieving fast convergence to global optimum given a sufficiently small dual step size η.

4.2 Greedy Column Generation

The Randomized AL-BCD (Algorithm 1), without knowledge of which columns are active, needs to update all M columns, and thus requires $O(MN)$ time for each outer iteration. In this section, we show how to utilize the decomposition in section 3 to efficiently find active columns and reduce the $O(MN)$ complexity.

Based on (18), we know, for a currently inactive column j with $W^{(t,s)}_{:,j} = 0$, the columnwise minimization (13) results in $\boldsymbol{d}^*_j \neq 0$ if and only if

$$\sum_{i=1}^{N} \left[-\nabla_{ij} f(W)\right]_+ > \lambda. \tag{19}$$

Therefore, we can use

$$s_j = \sum_{i=1}^{N} \left[-\nabla_{ij} f(W)\right]_+ \tag{20}$$

as criteria to select columns most likely to become nonzeros. Denote $\boldsymbol{s} \in \mathbb{R}^M$ as an M-dimensional vector recording scores (20) of the M columns. Computing $\boldsymbol{s}$ directly is expensive and requires $O(MNr)$ cost and $O(MN)$ memory. In the following, we propose a randomized greedy oracle that returns good columns with significant probability. Firstly, without the $[.]_+$ operator in (20), one can efficiently compute $\boldsymbol{s}$ by matrix-vector product

$$\begin{aligned} \boldsymbol{s} &= (-\nabla f(W))^T \mathbf{1} \\ &= (-\rho \boldsymbol{r}\mathbf{1}^T - \mathbb{D})^T \mathbf{1} = -\rho \mathbf{1}(\boldsymbol{r}^T \mathbf{1}) - V(U^T \mathbf{1}), \end{aligned} \tag{21}$$

which requires only cost $O(Mr + Nr)$ where r is the rank of $\mathbb{D}$. However, things become complicated when taking $[.]_+$ operation into account. First, the score for each column (20) can be written as

$$s_j = (-\nabla_j f(W))^T \boldsymbol{q}^j \tag{22}$$

where $\boldsymbol{q}^j$ is a vector determined by the signs of $-\nabla_j f(W)$, defined as

$$q_i^j = \begin{cases} 0 & , -\nabla_{ij} f(W) \leq 0 \\ 1 & , -\nabla_{ij} f(W) > 0 \end{cases} \tag{23}$$

The computation of (20) is harder than (21) in the sense that $\boldsymbol{q}_j$ is different for each column. Therefore, one cannot compute the whole vector $\boldsymbol{s}$ via a single matrix-vector product operation as in (21).

However, instead of requiring an exact oracle that finds a column with maximum score, we can employ a less expensive oracle that *returns columns of higher score with higher probability.* This can be achieved by uniformly sampling R vectors $\{\tilde{\boldsymbol{q}}_r\}_{r\in[R]}$ from the set $\{\boldsymbol{q}_j\}_{j=1}^M$ and compute the approximate score $\tilde{s}_j$ as

$$\tilde{s}_j = \max_{r\in[R]} (-\nabla_j f(W))^T \tilde{\boldsymbol{q}}_r. \tag{24}$$

Note that we take maximum in (24) because, for any vector $\tilde{\boldsymbol{q}}$ with elements $\tilde{q}_i \in \{0, 1\}$, we have

$$(-\nabla_j f(W))^T \tilde{\boldsymbol{q}} \leq (-\nabla_j f(W))^T \boldsymbol{q}^j = s_j.$$

Therefore the approximate score (24) is always an underestimate of the true score. By taking maximum (24), we are getting better estimate as R increases. In the next section, we will show that the approximate column-generation method actually leads to fast convergence to global optimum with a rate determined by R. In our experiments, setting $R = 10$ results in fast enough convergence empirically.

We summarize the AL-BCD with Column Generation in Algorithm 2. Note each iteration of Algorithm 2 performs only one step of column generation before each update of dual variables. As a technical contribution of this paper, we show in next section that such inexact minimization suffices for fast convergence.

5 Convergence Analysis

In this section, we analyze convergence of our Algorithm 1 and 2. Note existing analysis of Augmented Lagrangian Method (ALM) either requires solving each AL subproblem (10) to certain precision, resulting in a double-loop algorithm [28, 29, 30], or requires minimizing the function w.r.t. all blocks of variables in each iteration as in Alternating Direction Method of Multiplier (ADMM) [11]. As one technical contribution of this work, our analysis shows that a single-loop ALM with one step of (approximate) greedy block minimization per iteration suffices for global linear convergence.

Note that the AL sub-problem (10) does not satisfy strong convexity as usually required for proving linear convergence. However, it has strong convexity when restricted to a constant subspace. This restricted version of strong convexity has been exploited recently for proving linear-type of convergence [31, 33]. Utilizing such structure, we are able to show that both Randomized and Greedy BCD on the AL subproblem (10) have geometrically fast convergence as in the following theorems.

Theorem 3 (Linear Convergence of Randomized BCD)**.** *The iterate $\{\boldsymbol{x}^s\}_{s=1}^\infty$ produced by inner loop of Algorithm 1 has*

$$\mathbb{E}[f(W^{s+1})] - f^* \leq \left(1 - \frac{1}{M\gamma}\right)(f(W^s) - f^*).$$

where f^* is the optimum of (10)*,*

$$\gamma = \max\left\{16\rho\theta(f^0 - f^*)\ ,\ 2\theta(1 + 4L_g^2)\ ,\ 6\right\},$$

L_g is local Lipschitz-continuous constant of augmented term, and θ is Hoffman constant of the optimal (polyhedral) solution set.

Theorem 4 (Linear Convergence of Approximate Greedy BCD)**.** *Let $\{\boldsymbol{x}^s\}_{s=1}^\infty$ denote iterates produced by Algorithm 2 (without step 2.5) with a fixed $\boldsymbol{\alpha}$. Then*

$$\mathbb{E}[f(W^{s+1})] - f^* \leq \left(1 - \frac{1}{m\gamma_2}\right)(f(W^s) - f^*),$$

where $m = M/R$,

$$\gamma_2 = \max\left\{16\rho\theta_1(f^0 - f^*)\ ,\ 2\theta_1(1 + 4L_g^2)\ ,\ 6\right\},$$

and θ_1 is the $\ell_{2,1}$-norm version of Hoffman constant satisfying $\theta \leq \theta_1 \leq M\theta$.

Then we further show that, for a small enough constant step size η, Algorithm 1 and 2 have linear convergence to the optimum. Let $d(\boldsymbol{\alpha}) = \min_{W\geq 0}\ \mathcal{L}(W, \boldsymbol{\alpha})$ be the augmented dual objective of (3), $d^* := \max_{\boldsymbol{\alpha}}\ d(\boldsymbol{\alpha})$ be the optimal dual objective and let $\Delta_d^t := d^* - d(\boldsymbol{\alpha}^t)$, $\Delta_p^t := \mathcal{L}(W^{t+1}, \boldsymbol{\alpha}^t) - d(\boldsymbol{\alpha}^t)$ be the dual and primal suboptimality respectively.

Theorem 5 (Linear Convergence of Randomized ALBCD)**.** *The iterates $\{(W^t, \boldsymbol{\alpha}^t)\}_{t=1}^{\infty}$ produced by Algorithm 1 has*

$$\Delta_d^t + \Delta_p^t \leq \frac{1}{1+\min(\frac{1}{2M\gamma}, \frac{\eta}{\tau})} \left(\Delta_d^{t-1} + \Delta_p^{t-1}\right),$$

for any $0 < \eta \leq \rho/4M\gamma$, where $\tau > 0$ is a constant depending on the geometry of optimal solution set.

Theorem 6 (Linear Convergence of Greedy ALBCD)**.** *The iterates $\{(W^t, \boldsymbol{\alpha}^t)\}_{t=1}^{\infty}$ produced by Algorithm 2 has*

$$\Delta_d^t + \Delta_p^t \leq \frac{1}{1+\min(\frac{1}{2m\gamma_2}, \frac{\eta}{\tau})} \left(\Delta_d^{t-1} + \Delta_p^{t-1}\right),$$

for any $0 < \eta \leq \rho/4m\gamma_2$.

6 Experiments

6.1 Scalable Exemplar Clustering

In this section, we compare the proposed AL-BCD approaches to existing methods for convex exemplar clustering (3) with normalized square Euclidean distance, which is a special case of the Bregman Divergence (4). In particular, the methods in comparison are listed below.

- **ADMM**: The Alternating Direction Method of Multiplier (ADMM) solver proposed in [6, 35].
- **LP-I**: The Interior-Point-Method (IPM) Linear Programming (LP) solver in Matlab.
- **LP-A**: The Augmented-Lagrangian Coordinate Descent (AL-CD) LP solver recently proposed in [36], which demonstrated better efficiency compared to IPM and Primal, Dual Simplex methods implemented in state-of-the-art commercial solver *CPLEX* on large-scale problems.[1]
- **ALRCD**: The proposed Augmented Lagrangian Method with randomized Block Coordinate Descent (Algorithm 1).
- **ALGCD**: The proposed AL-BCD method with Greedy Column Generation (Algorithm 2).

Note that we did not compare with Subgradient-based LP solver [3] since they can hardly obtain solution of reasonable precision to be compared with other methods. Note also that message-passing algorithms such as Affinity-Propagation (AP) and Max-Product Linear Programming (MPLP) are also excluded from comparison since they do not guarantee convergence to optimum of the convex program, and they also require construction of MN dissimilarity matrix and thus can only be used in small-scale problems [8, 16].

The timing results are shown in Table 1, where we divide the table into 4 blocks from top to bottom. The first 3 blocks are data with increasing number of samples N $(= M)$ and the 4th block is data with nonlinear random features. The timing result is marked with "n/a" if the time required is more than 8 hours or the memory required is more than 100GB.

On small-size data set, it is clear that LP-I is the slowest and even for data with $N \approx 500$ it takes 6 hours to run. On the other hand, we observe that LP-A is an order of magnitude faster than ADMM, and ALBCD is another order of magnitude faster than LP-A on medium-scale data. Note this is expected since both LP-A and ALRCD performs (block) coordinate descent on the Augmented Lagrangian problem, but ALRCD has closed-form solution for each column, which makes it converge much faster than LP-A. On large-scale data sets, both ADMM and LP-A become infeasible and even storing the dissimilarity matrix requires tens of gigabytes. In this scenario, the column-generation-based solver ALGCD is the only efficient solver and is another order of magnitude faster than pure ALRCD. The last 3 data sets in Table 1 demonstrate the usage of our proposed method together with a Random-Feature Kernel Approximation method [26, 34] to perform exemplar clustering with (RBF-Laplacian) Kernelized Square Euclidean Distance, where using the Random Binning Feature proposed in [26], we obtain a decomposition of dissimilarity matrix $\mathbb{D} = UV^T$ of U, V that are sparse but not low-rank.

In Table 2, 3, we test scalability of different methods on subsamples of Covtype data set with increasing number of samples, where we can observe that, for LP and ALRCD, the time and space required to solve (3) grows quadratically with N, while ALGCD has running time growing only slightly superlinear to N and memory consumption growing linear to N (due to a fixed-size cache of columns).

6.2 Facility Location

In this section, we demonstrate the usage of our proposed method on the Facility Location problem. We use the World TSP data set which contains the latitude and longitude of 1,904,711 cities [2]in the world to model the location of demands (customers) and services (facilities). In particular, we randomly sampled $N = 1,000,000$ cities as demand locations and the re-

[1] The solver can be downloaded at: `http://www.cs.utexas.edu/~ianyen/LPsparse`.

Table 1: Timing results of Convex Exemplar Clustering solvers with normalized square Euclidean distance, where d_{avg} is the average number of non-zero features per sample, K is the number of clusters obtained from solving (3) with $\lambda \in [0.01N, 0.1N]$ chosen to induce integer solution (if any between the range). A "*" mark on "K" means the solution is integer (and thus is optimal to the combinatorial problem (1)).

Data	**N (M)**	**d**	$\mathbf{d_{avg}}$	**Size($\mathbb{D}$)**	**K**	**ADMM**	**LP-I**	**LP-A**	**ALRCD**	**ALGCD**
Iris	150	4	4	0.42MB	11	11.23s	1m40s	0.43s	0.34s	**0.105s**
Wine	178	13	13	0.58MB	16	5.12s	4m34s	0.68s	0.37s	**0.052s**
Glass	214	9	9	0.87MB	6^*	2m7s	8m39s	1.20s	0.41s	**0.079s**
Vowel	528	10	10	5MB	25	4m32s	6h14m	24.6s	10.29s	**2.24s**
Scene	1211	294	294	28MB	2^*	2h27m	n/a	2m4s	**5.64s**	7.06s
Satimage	4435	36	36	355MB	7^*	n/a	n/a	6h29m	3m3s	**17.5s**
Sector	6412	55197	163	738M	6^*	n/a	n/a	n/a	15m2s	**1m36s**
Pendigit	7494	16	16	1.1GB	7^*	n/a	n/a	n/a	5m51s	**2.55s**
RCV1	20242	47236	74	6.6GB	9	n/a	n/a	n/a	5h21m	**1m44s**
CodRNA	59535	8	8	53GB	7^*	n/a	n/a	n/a	7h32m	**1m41s**
Covtype	581012	54	11.94	>100G	7^*	n/a	n/a	n/a	n/a	**35m58s**
$\mathbf{Pendigit_{RF}}$	7494	12891	100	1.1GB	12	n/a	n/a	n/a	6m35s	**36.5s**
$\mathbf{CodRNA_{RF}}$	59535	7611	50	53GB	5	n/a	n/a	n/a	1h46m	**1m4s**
$\mathbf{Covtype_{RF}}$	581012	54509	50	>100G	11	n/a	n/a	n/a	n/a	**44m38s**

Data size	K	LP-A	ALRCD	ALGCD
$N = 5 \times 10^3$	13	7h11s	3m34s	**2.7s**
$N = 5 \times 10^4$	14	n/a	6h36m	**1m53s**
$N = 5 \times 10^5$	14	n/a	n/a	**28m14s**

Table 2: Timing results on *Covtype* data set ($d = 55$, $M = N$, $\lambda = 0.01N$).

Data size	K	LP-A	ALRCD	ALGCD
$N = 5 \times 10^3$	13	5.4GB	0.2GB	**23MB**
$N = 5 \times 10^4$	14	n/a	19GB	**220MB**
$N = 5 \times 10^5$	14	n/a	n/a	**2.1GB**

Table 3: Memory consumption on *Covtype* data set ($d = 55$, $M = N$, $\lambda = 0.01N$), where ALGCD (with low-rank column generation) uses a LRU cache of size=500 columns.

Data size	ALRCD(sub)	ALRCD	ALGCD
$N = 10^4$	1h11m	1h54m	**1m20s**
$M = 10^4$	obj=3,630	**3,550**	3,567
$N = 10^6$	n/a	n/a	**1h58m**
$M \approx 10^6$	n/a	n/a	obj=342,273

Table 4: Facility Location on 1,904,711 cities over the worlds ($\lambda = 0.005N$).

maining $M = 904,711$ cities as potential positions for building facilities. We then use Geographical distance (computed via polar coordinate flat-Earth formula) to measure the distance between customer and potential facility. To obtain a low-rank approximation to N by M distance matrix $\mathbb{D}$, we sampled $100N$ pairs of distance and use Matrix Completion [3]to find $UV^T \approx \mathbb{D}$. Using rank $r = 40$, we obtain a decomposition UV^T with testing RMSE $< 10^{-3}$.

Table 4 compares results obtained from (i) ALBCD running with 10% sub-sampled columns, (ii) ALBCD using exact matrix $\mathbb{D}$ of Geographical distance, and (iii) ALBCD-CG using low-rank approximation $\mathbb{D} = UV^T$ from matrix completion with $r = 40$.

In the table, only ALBCD exactly solves problem (3), while the other two only solve it approximately. However, the approximation made by ALBCD-CG based on approximate low-rank decomposition gets much lower objective than that from column-subsampling approach. On the other hand, the time required by ALBCD-CG is much less than the others. Finally, on full data set with $N = 10^6$ (10^{12} #variables), ALBCD-CG is the only feasible way to solve (3), which takes about 2 hours—a time increase almost linear to N.

6.3 Active learning via Exemplar clustering

In supervised classification, one often has access to a large amount of unlabeled data, and a significant effort is needed to provide labels for the data (hiring annotators, domain experts, or crowd-sourcing via the mechanical turk). The goal of active learning is to judiciously select a small subset of examples to label in order to maximize the improvement in classification accuracy over unseen test samples.

Many popular active learning approaches make use of classifier uncertainty scores of the unlabeled samples to select a subset for labeling. The uncertainty score

[3] The data set can be downloaded at `http://www.math.uwaterloo.ca/tsp/world/`.

[3] We use Matrix Completion solver provided by the authors of [37].

represents the lack of confidence that the classifier has in its top class, and serves as a proxy for expected improvement in classification accuracy by labeling this example. Studies have also shown that encouraging diversity in the actively selected batch of samples can further help in improving the classifier performance with reduced labeling effort [4, 7, 32]. Exemplar clustering provides a very appealing mechanism to implement such a scheme for active learning. Uncertainty scores can be used to modulate the regularization parameter λ (Eq. 3) such that highly uncertain samples are penalized less. This has balances the two desirable properties in the selected batch – diversity and uncertainty. We demonstrate that the proposed ALBCD approach is highly suitable for medium to large scale active learning whereas previous approaches tend to be slower by an order of magnitude [5, 7].

We experiment with Ohio State University Medical (OHSUMED) text data[4] [24] which consists of about 35K medical abstracts from 23 cardiovascular diseases categories. For simplicity, we operate under a multiclass classification setting and ignore the abstracts having more than one label. This leaves us with $18,302$ abstracts which we split in 2:1 ratio for training and test, respectively. We use cosine distance between the tf-idf representation of the abstacts as the dissimilarity measure, which is a popular distance measure used for text documents. It also fits well with our approach, as the tf-idf representations are typically extremely sparse, thus allowing fast matrix-vector products, without the need to explicitly construct the full dissimilarity matrix. We use one-vs-rest multiclass SVM as the classifier.

We compare three approaches for incremental selection of examples for labeling: *random sampling*, *uncertainty sampling*, *uncertainty sampling with diversity*. For *random sampling* and *uncertainty sampling*, the initial classifier is trained with a randomly selected batch of 100 examples. For the third approach that combines uncertainty sampling with diversity, we use exemplar clustering to select the initial batch of 100 examples for training the classifier. For all approaches, we incrementally add labeled examples in the batches of 200 each until we reach 1500 examples. Figure 1 shows the plots of accuracy on the test set against the number of training examples for all the approaches. Active learning approach balancing uncertainty and diversity indeed improves classification accuracy especially in the early periods with less labeled examples. For selecting one batch of 100 exemplars out of $12,192$ training samples, the proposed ALBCD approach takes about 5 minutes while the ADMM based approach [5] takes more than an hour.

[4] `http://disi.unitn.it/moschitti/corpora.htm`

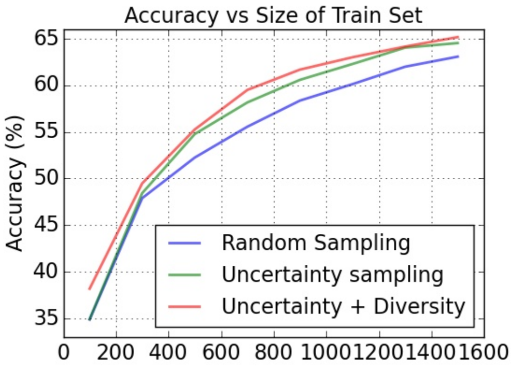
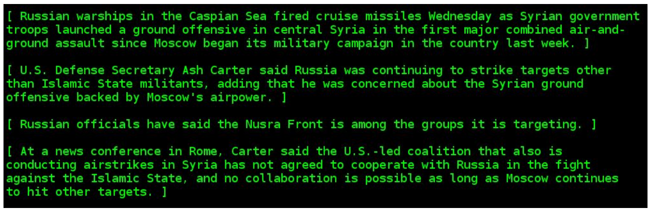


Figure 1: Learning curve for OHSUMED dataset.

6.4 Document summarization

Exemplar clustering also provides a simple but surprisingly effective tool for document summarization. The goal here is to select a short snippet of text that summarizes the main points made across several documents, such as news articles. Often the snippets are selected as a few representative sentences from the documents. Here we illustrate that convex exemplar clustering based on ALBCD can also be applied to large-scale document summarization. We note that to define a distance matrix (dissimilarity matrix) between the sentences, one can use either the sparse tf-idf representation or the low-rank mean-word2vec representation [22], where each sentence is simply represented as an average of the word2vec vectors for each of the words. We illustrate the summaries produced by ALBCD from 10 news articles from CNN, Reuters e.t.c. that describe the recent Russian rocket launches in Syria in Figure 2. Qualitatively, the 4 chosen sentences are not redundant, and each one carries significant topical information (as measured by fraction of topic-specific words). We also illustrate the scalability of the approach by applying it to 20,000 sentences from the Reuters RCV1 dataset. We have used a 300-dimensional vector-space representation of each sentence by computing the average word2vec vector over the words in a sentence. It takes 3 minutes to compute 200 representative exemplars from this collection.



```
[ Russian warships in the Caspian Sea fired cruise missiles Wednesday as Syrian government
troops launched a ground offensive in central Syria in the first major combined air-and-
ground assault since Moscow began its military campaign in the country last week. ]

[ U.S. Defense Secretary Ash Carter said Russia was continuing to strike targets other
than Islamic State militants, adding that he was concerned about the Syrian ground
offensive backed by Moscow's airpower. ]

[ Russian officials have said the Nusra Front is among the groups it is targeting. ]

[ At a news conference in Rome, Carter said the U.S.-led coalition that also is
conducting airstrikes in Syria has not agreed to cooperate with Russia in the fight
against the Islamic State, and no collaboration is possible as long as Moscow continues
to hit other targets. ]
```

Figure 2: Summary sentences automatically selected using convex exemplar clustering from 10 news articles about the russian missile launches in Syria.

References

[1] A. A. Ageev and M. I. Sviridenko. An 0.828-approximation algorithm for the uncapacitated facility location problem. *Discrete Applied Mathematics*, 93(2):149–156, 1999.

[2] A. Badanidiyuru, B. Mirzasoleiman, A. Karbasi, and A. Krause. Streaming submodular maximization: Massive data summarization on the fly. In *Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 671–680. ACM, 2014.

[3] F. Barahona and F. Chudak. Near-optimal solutions to large-scale facility location problems. *Discrete Optimization*, 2005.

[4] K. Brinker. Incorporating diversity in active learning with support vector machines. In *ICML*, volume 3, pages 59–66, 2003.

[5] E. Elhamifar, G. Sapiro, and S. S. Sastry. Dissimilarity-based sparse subset selection. *arXiv preprint arXiv:1407.6810*, 2014.

[6] E. Elhamifar, G. Sapiro, and R. Vidal. Finding exemplars from pairwise dissimilarities via simultaneous sparse recovery. In *Advances in Neural Information Processing Systems*, pages 19–27, 2012.

[7] E. Elhamifar, G. Sapiro, A. Yang, and S. S. Sasrty. A convex optimization framework for active learning. In *Computer Vision (ICCV), 2013 IEEE Int. Conf. on*. IEEE, 2013.

[8] B. J. Frey and D. Dueck. Clustering by passing messages between data points. *science*, 2007.

[9] P. Hansen, J. Brimberg, D. Urosevic, and N. Mladenovic. Primal-dual variable neighborhood search for the simple plant-location problem. *INFORMS Journal on Computing*, 19(4):552–564, 2007.

[10] A. Hoffman. On approximate solutions of systems of linear inequalities. *Journal of Research of the National Bureau of Standards*, 1952.

[11] M. Hong and Z.-Q. Luo. On the linear convergence of the alternating direction method of multipliers. *arXiv preprint arXiv:1208.3922*, 2012.

[12] C.-J. Hsieh and P. Olsen. Nuclear norm minimization via active subspace selection. In *Proceedings of the 31st International Conference on Machine Learning (ICML-14)*, pages 575–583, 2014.

[13] L. Kaufman and P. Rousseeuw. *Clustering by means of medoids*. North-Holland, 1987.

[14] A. Kulesza and B. Taskar. Determinantal point processes for machine learning. *Foundations and Trends in Machine Learning*, 2012.

[15] D. Lashkari and P. Golland. Convex clustering with exemplar-based models. In *Advances in neural information processing systems*, 2007.

[16] N. Lazic, B. J. Frey, and P. Aarabi. Solving the uncapacitated facility location problem using message passing algorithms. In *International Conference on Artificial Intelligence and Statistics*, 2010.

[17] M. Li, J. T. Kwok, and B.-L. Lu. Making large-scale nyström approximation possible. In *ICML 2010-Proceedings, 27th International Conference on Machine Learning*, 2010.

[18] S. Li. A 1.488 approximation algorithm for the uncapacitated facility location problem. *Information and Computation*, 2013.

[19] S. Li and O. Svensson. Approximating k-median via pseudo-approximation. In *proceedings of the forty-fifth annual ACM symposium on theory of computing*. ACM, 2013.

[20] H. Liu, M. Palatucci, and J. Zhang. Blockwise coordinate descent procedures for the multi-task lasso, with applications to neural semantic basis discovery. In *Proceedings of the 26th Annual International Conference on Machine Learning*. ACM, 2009.

[21] W. Liu, J. Wang, S. Kumar, and S.-F. Chang. Hashing with graphs. In *Proceedings of the 28th international conference on machine learning (ICML-11)*, 2011.

[22] T. Mikolov, I. Sutskever, K. Chen, G. Corrado, and J. Dean. Distributed representations of words and phrases and their compositionality. In *Advances in neural information processing systems*, pages 3111–3119, 2013.

[23] B. Mirzasoleiman, A. Karbasi, R. Sarkar, and A. Krause. Distributed submodular maximization: Identifying representative elements in massive data. In *Advances in Neural Information Processing Systems*, pages 2049–2057, 2013.

[24] A. Moschitti and R. Basili. Complex linguistic features for text classification: A comprehensive study. In *Advances in Information Retrieval*, pages 181–196. Springer, 2004.

[25] A. Nellore and R. Ward. Recovery guarantees for exemplar-based clustering. *arXiv preprint arXiv:1309.3256*, 2013.

[26] A. Rahimi and B. Recht. Random features for large-scale kernel machines. In *Advances in neural information processing systems*, pages 1177–1184, 2007.

[27] B. Recht, M. Fazel, and P. A. Parrilo. Guaranteed minimum-rank solutions of linear matrix

equations via nuclear norm minimization. *SIAM review*, 52(3):471–501, 2010.

[28] R. T. Rockafellar. Augmented lagrangians and applications of the proximal point algorithm in convex programming. *Mathematics of operations research*, 1976.

[29] R. T. Rockafellar. Monotone operators and the proximal point algorithm. *SIAM journal on control and optimization*, 1976.

[30] R. Tomioka, T. Suzuki, and M. Sugiyama. Super-linear convergence of dual augmented lagrangian algorithm for sparsity regularized estimation. *The Journal of Machine Learning Research*, 2011.

[31] P.-W. Wang and C.-J. Lin. Iteration complexity of feasible descent methods for convex optimization. *The Journal of Machine Learning Research*, 2014.

[32] Y. Yang, Z. Ma, F. Nie, X. Chang, and A. G. Hauptmann. Multi-class active learning by uncertainty sampling with diversity maximization. *International Journal of Computer Vision*, 113(2):113–127, 2014.

[33] I. E.-H. Yen, C.-J. Hsieh, P. K. Ravikumar, and I. S. Dhillon. Constant nullspace strong convexity and fast convergence of proximal methods under high-dimensional settings. In *Advances in Neural Information Processing Systems*, 2014.

[34] I. E.-H. Yen, T.-W. Lin, S.-D. Lin, P. K. Ravikumar, and I. S. Dhillon. Sparse random feature algorithm as coordinate descent in hilbert space. In *Advances in Neural Information Processing Systems*, pages 2456–2464, 2014.

[35] I. E.-H. Yen, X. Lin, E. K. Zhong, P. Ravikumar, and I. S. Dhillon. A convex exemplar-based approach to mad-bayes dirichlet process mixture models. In *Proceedings of The 32nd International Conference on Machine Learning*, 2015.

[36] I. E.-H. Yen, K. Zhong, C.-J. Hsieh, P. K. Ravikumar, and I. S. Dhillon. Sparse linear programming via primal and dual augmented coordinate descent. In *Advances in Neural Information Processing Systems*, pages 2359–2367, 2015.

[37] H.-F. Yu, C.-J. Hsieh, S. Si, and I. S. Dhillon. Scalable coordinate descent approaches to parallel matrix factorization for recommender systems. In *IEEE International Conference of Data Mining*, 2012.