Proceedings of the Twenty-Ninth AAAI Conference on Artificial Intelligence

10,000+ Times Accelerated Robust Subset Selection (ARSS)

Feiyun Zhu, Bin Fan, Xinliang Zhu, Ying Wang, Shiming Xiang and Chunhong Pan
Institute of Automation, Chinese Academy of Sciences
{fyzhu, bfan, ywang, smxiang and chpan}@nlpr.ia.ac.cn, zhuxinliang2012@ia.ac.cn

Abstract

Subset selection from massive data with noised information is increasingly popular for various applications. This problem is still highly challenging as current methods are generally slow in speed and sensitive to outliers. To address the above two issues, we propose an accelerated robust subset selection (ARSS) method. Specifically in the subset selection area, this is the first attempt to employ the ℓ_p $(0 < p \leq 1)$-norm based measure for the representation loss, preventing large errors from dominating our objective. As a result, the robustness against outlier elements is greatly enhanced. Actually, data size is generally much larger than feature length, i.e. $N \gg L$. Based on this observation, we propose a speedup solver (via ALM and equivalent derivations) to highly reduce the computational cost, theoretically from $O\left(N^4\right)$ to $O\left(N^2 L\right)$. Extensive experiments on ten benchmark datasets verify that our method not only outperforms state of the art methods, but also runs 10,000+ times faster than the most related method.

Introduction

Due to the explosive growth of data (Wang, Kumar, and Chang 2012), subset selection methods are increasingly popular for a wide range of machine learning and computer vision applications (Frey and Dueck 2007; Jenatton, Audibert, and Bach 2011). This kind of methods offer the potential to select a few highly representative samples or exemplars to describe the entire dataset. By analyzing a few, we can roughly know all. Such case is very important to summarize and visualize huge datasets of texts, images and videos etc. (Bien and Tibshirani 2011; Elhamifar et al. 2012b). Besides, by only using the selected exemplars for succeeding tasks, the cost of memories and computational time will be greatly reduced (Garcia et al. 2012). Additionally, as outliers are generally less representative, the side effect of outliers will be reduced, thus boosting the performance of subsequent applications (Elhamifar et al. 2012a).

There have been several subset selection methods. The most intuitional method is to randomly select a fixed number of samples. Although highly efficient, there is no guarantee for an effective selection. For the other methods, depending on the mechanism of representative exemplars, there are mainly three categories of selection methods. One category

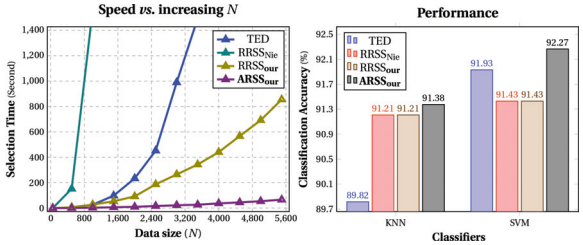

Figure 1: Comparisons of four algorithms on Optdigit. Two conclusions can be drawn. First, our method (ARSS_{our}) is highly faster than all others; with the help of an elegant new theorem, RRSS_{our} is significantly faster than the authorial algorithm RRSS_{Nie}. Second, ARSS_{our} achieves highly promising prediction accuracies.

relies on the assumption that the data points lie in one or multiple low-dimensional subspaces. Specifically, the Rank Revealing QR (RRQR) (Chan 1987; Boutsidis, Mahoney, and Drineas 2009) selects the subsets that give the best conditional sub-matrix. Unfortunately, this method has suboptimal properties, as it is not assured to find the globally optimum in polynomial time.

Another category assumes that the samples are distributed around centers (Frey and Dueck 2007; Liu et al. 2010). The center or its nearest neighbour are selected as exemplars. Perhaps, Kmeans and Kmedoids are the most typical methods (Kmedoids is a variant of Kmeans). Both methods employ an EM-like algorithm. Thus, the results depend tightly on the initialization, and they are highly unstable for large K (i.e. the number of centers or selected samples).

Recently, there are a few methods that assume exemplars are the samples that can best represent the whole dataset. However, for (Yu, Bi, and Tresp 2006), the optimization is a combinatorial problem (NP-hard) (Nie et al. 2013; Yu et al. 2008), which is computationally intractable to solve. Besides, the representation loss is measured by the least square measure, which is sensitive to outliers in data (Wang et al. 2014; Zhu et al. 2014; Nie et al. 2013).

Then (Nie et al. 2013) improves (Yu, Bi, and Tresp 2006) by employing a robust loss via the $\ell_{2,1}$-norm; the ℓ_1-norm is applied to samples, and the ℓ_2-norm is used for features. In this way, the side effect of outlier samples is relieved. The

solver of (Nie et al. 2013) is theoretically perfect due to its ability of convergence to global optima. Unfortunately, in terms of computational costs, the solver is highly complex. It takes $O\left(N^4\right)$ for one iteration as shown in Table 1. This is infeasible for the case of large N (e.g. it takes 2000+ hours for a case of $N = 13000$). Moreover, the representation loss is only robust against outlier samples. Such case is worth improvement, as there may exist outlier elements in real data.

Contributions. In this paper, we propose an accelerated robust subset selection method to highly raise the speed on the one hand, and to boost the robustness on the other. To this end, we use the $\ell_p\,(0<p\leq 1)$-norm based robust measure for the representation loss, preventing large errors from dominating our objective. As a result, the robustness against outliers is greatly boosted. Then, based on the observation that data size is generally much larger than feature length, i.e. $N \gg L$, we propose a speedup solver. The main acceleration is owing to the Augmented Lagrange Multiplier (ALM) and an equivalent derivation. Via them, we reduce the computational complexity from $O\left(N^4\right)$ to $O\left(N^2L\right)$. Extensive results on ten benchmark datasets demonstrate that in average, our method is 10,000+ times faster than Nie's method. The selection quality is highly encouraging as shown in Fig. 1. Additionally, via another equivalent derivation, we give an accelerated solver for Nie's method, theoretically reducing the computational complexity from $O\left(N^4\right)$ to $O\left(N^2L + NL^3\right)$ as listed in Table 1, empirically obtaining a 500+ times speedup compared with the authorial solver.

Notations. We use boldface uppercase letters to denote matrices and boldface lowercase letters to represent vectors. For a matrix $\mathbf{Y} = [Y_{ln}] \in \mathbb{R}^{L\times N}$, we denote its l^{th} row and n^{th} column as $\mathbf{y}^l$ and $\mathbf{y}_n$ respectively. The $\ell_{2,1}$-norm of a matrix is defined as $\|\mathbf{Y}\|_{2,1} = \sum_l^L \sqrt{\sum_n^N Y_{ln}^2} = \sum_l^L \left\|\mathbf{y}^l\right\|_2$. The $\ell_p\,(0 < p \leq 1)$-norm of a matrix is defined as $\|\mathbf{Y}\|_p = \left(\sum_n^N \sum_l^L |Y_{ln}|^p\right)^{\frac{1}{p}}$; thus, we have $\|\mathbf{Y}\|_p^p = \sum_{l,n} |Y_{ln}|^p$.

Subset Selection via Self-Representation

In the problem of subset selection, we are often given a set of N unlabelled points $\mathbf{X} = \left\{\mathbf{x}_1, \mathbf{x}_2, \cdots, \mathbf{x}_N \,|\, \mathbf{x}_n \in \mathbb{R}^L\right\}$, where L is the feature length. The goal is to select the top $K\,(K \ll N)$ most representative and informative samples (i.e. exemplars) to effectively describe the entire dataset $\mathbf{X}$. By solely using these K exemplars for subsequent tasks, we could greatly reduce the computational costs and largely alleviate the side effects of outlier elements in data. Such a motivation could be formulated as the Transductive Experimental Design (TED) model (Yu, Bi, and Tresp 2006):

$$\min_{\mathbf{Q},\mathbf{A}} \sum_{n=1}^{N} \left(\|\mathbf{x}_n - \mathbf{Q}\mathbf{a}_n\|_2^2 + \alpha \|\mathbf{a}_n\|_2^2\right), \qquad (1)$$

where $\mathbf{Q} \in \mathbb{R}^{L\times K}$ is the selected subset matrix, whose column vectors all come from $\mathbf{X}$, i.e. $\mathbf{q}_k \in \mathbf{X}, \forall k \in \{1, \cdots, K\}$; $\mathbf{A} = [\mathbf{a}_1, \cdots, \mathbf{a}_N] \in \mathbb{R}^{K\times N}$ is the corresponding linear combination coefficients. By minimizing (1), TED could select the highly informative and representative samples, as they have to well represent all the samples in $\mathbf{X}$.

Table 1: Complexity comparison of three algorithms at one iteration step. Generally, data size is much larger than feature length, i.e. $N \gg L$. Compared with RRSS_{Nie} (*Nie's model via the authorial solver*), RRSS_{our} (*Nie's method speeded up by our solver*l) and ARSS_{our} (*ours*) are significantly simplified.

Methods	RRSS_{Nie}	RRSS_{our}	ARSS_{our}
Complex.	$O\left(N^4\right)$	$O\left(N^2L + NL^3\right)$	$O\left(N^2L\right)$

Although TED (1) is well modeled—very accurate and intuitive, there are two bottlenecks. First, the objective is a combinatorial optimization problem. It is NP-hard to exhaustively search the optimal subset $\mathbf{Q}$ from $\mathbf{X}$. For this reason, the author approximate (1) via a sequential optimization problem, which is solved by an inefficient greedy optimization algorithm. Second, similar to the existing least square loss based models in machine learning and statistics, (1) is sensitive to the presence of outliers (Wang et al. 2014).

Accordingly, Nie *et al.* propose a new model (RRSS):

$$\min_{\mathbf{A}\in\mathbb{R}^{N\times N}} \sum_{n=1}^{N} \|\mathbf{x}_n - \mathbf{X}\mathbf{a}_n\|_2 + \gamma \|\mathbf{A}\|_{2,1}\,, \qquad (2)$$

where γ is a nonnegative parameter; $\mathbf{A}$ is constrained to be row-sparse, and thus to select the most representative and informative samples (Nie et al. 2013). As the representation loss is accumulated via the ℓ_1-norm among samples, compared with (1), the robustness against outlier samples is enhanced. Equivalently, (2) is rewritten in the matrix format:

$$\min_{\mathbf{A}\in\mathbb{R}^{N\times N}} \left\|(\mathbf{X} - \mathbf{X}\mathbf{A})^T\right\|_{2,1} + \gamma \|\mathbf{A}\|_{2,1}\,. \qquad (3)$$

Since the objective (3) is convex in $\mathbf{A}$, the global minimum may be found by differentiating (3) and setting the derivative to zero (Levin et al. 2008), resulting in a linear system[1]

$$\mathbf{a}_n = U_{nn}\left(U_{nn}\mathbf{X}^T\mathbf{X} + \gamma\mathbf{V}\right)^{-1}\mathbf{X}^T\mathbf{x}_n,\ \forall n = \{1,2,\cdots,N\}, \quad (4)$$

where $\mathbf{V} \in \mathbb{R}^{N\times N}$ is a diagonal matrix with the n^{th} diagonal entry as $V_{nn} = \frac{1}{2\|\mathbf{a}^n\|_2}$ and $U_{nn} = \frac{1}{2\|\mathbf{x}_n - \mathbf{X}\mathbf{a}_n\|_2}$.

It seems perfect to use (4) to solve the objective (3), because (4) looks simple and the global optimum is theoretically guaranteed (Nie et al. 2013). Unfortunately, in terms of speed, (4) is usually infeasible due to the incredible computational demand in the case of large N (the number of samples). At each iteration, the computational complexity of (4) is up to $O\left(N^4\right)$, as analyzed in Remark 1. According to our experiments, the time cost is up to 2088 hours (i.e. 87 days) for a subset selection problem of 13000 samples.

[1] To avoid singular failures, we get $V_{nn} = \frac{1}{2\sqrt{\|\mathbf{a}^n\|_2^2 + \epsilon}}$, $U_{nn} = \frac{1}{2\sqrt{\|\mathbf{x}_n - \mathbf{X}\mathbf{a}_n\|_2^2 + \epsilon}}$ $(\epsilon > 0)$. Then the algorithm is to minimize the objective of $\sum_n^N \sqrt{\|\mathbf{x}_n - \mathbf{X}\mathbf{a}_n\|_2^2 + \epsilon} + \gamma \sum_n^N \sqrt{\|\mathbf{a}^n\|_2^2 + \epsilon}$. When $\epsilon \to 0$, this objective is reduced to the objective (3).

Remark 1. *Since $U_{nn}\mathbf{X}^T\mathbf{X}+\gamma\mathbf{V}\in\mathbb{R}^{N\times N}$, the major computational cost of* (4) *focuses on a $N\times N$ linear system. If solved by the Cholesky factorization method, it costs $\frac{1}{3}N^3$ for factorization as well as $2N^2$ for forward and backward substitution. This amounts to $O\left(N^3\right)$ in total. By now, we only solve $\mathbf{a}_n$. Once solving all the set of $\{\mathbf{a}_n\}_{n=1}^N$, the total complexity amounts to $O\left(N^4\right)$ for one iteration step.*

Accelerated Robust Subset Selection (ARSS)

Due to the huge computational costs, Nie's method is infeasible for the case of large N—the computational time is up to 2088 hours for a case of 13000 samples. Besides, Nie's model (3) imposes the ℓ_2-norm among features, which is prone to outliers in features. To tackle the above two issues, we propose a more robust model in the $\ell_p\,(0<p\leq 1)$-norm. Although the resulted objective is challenging to solve, a speedup algorithm is proposed to dramatically save the computational costs. For the same task of $N=13000$, it costs our method 1.8 minutes, achieving a 68429 times acceleration compared with the speed of Nie's method.

Modeling. To boost the robustness against outliers in both samples and features, we formulate the discrepancy between $\mathbf{X}$ and $\mathbf{XA}$ via the $\ell_p(0<p<1)$-norm. There are theoretical and empirical evidences to verify that compared with ℓ_2 or ℓ_1 norms, the ℓ_p-norm is more able to prevent outlier elements from dominating the objective, enhancing the robustness (Nie et al. 2012). Thus, we have the following objective

$$\min_{\mathbf{A}\in\mathbb{R}^{N\times N}} \mathcal{O}=\|\mathbf{X}-\mathbf{XA}\|_p^p+\gamma\|\mathbf{A}\|_{2,1}\,, \tag{5}$$

where γ is a balancing parameter; $\mathbf{A}$ is a row sparse matrix, used to select the most informative and representative samples. By minimizing the energy of (5), we could capture the most essential properties of the dataset $\mathbf{X}$.

After obtaining the optimal $\mathbf{A}$, the row indexes are sorted by the row-sum value of the absolute $\mathbf{A}$ in decreasing order. The samples specified by the top K indexes are selected as exemplars. Note that the model (5) could be applied to the unsupervised feature selection problem by only transposing the data matrix $\mathbf{X}$. In this case, $\mathbf{A}$ is a $L\times L$ row sparse matrix, used to select the most representative features.

Accelerated Solver for the ARSS Objective in (5)

Although objective (5) is challenging to solve, we propose an effective and highly efficient solver. The acceleration owes to the ALM and an equivalent derivation.

ALM The most intractable challenge of (5) is that, the $\ell_p\,(0<p\leq 1)$-norm is non-convex, non-smooth and not-differentiable at the zero point. Therefore, it is beneficial to use the Augmented Lagrangian Method (ALM) (Nocedal and Wright 2006) to solve (5), resulting in several easily tackled unconstrained subproblems. By solving them iteratively, the solutions of subproblems could eventually converge to a minimum (Li 2011; Meng et al. 2013).

Specifically, we introduce an auxiliary variable $\mathbf{E}=\mathbf{X}-\mathbf{XA}\in\mathbb{R}^{L\times N}$. Thus, the objective (5) becomes:

$$\min_{\mathbf{A},\mathbf{E}=\mathbf{X}-\mathbf{XA}} \|\mathbf{E}\|_p^p+\gamma\|\mathbf{A}\|_{2,1}\,. \tag{6}$$

To deal with the equality constraint in (6), the most convenient method is to add a penalty, resulting in

$$\min_{\mathbf{A}} \|\mathbf{E}\|_p^p+\gamma\|\mathbf{A}\|_{2,1}+\frac{\mu}{2}\|\mathbf{E}-\mathbf{X}+\mathbf{XA}\|_F^2\,, \tag{7}$$

where μ is a penalty parameter. To guarantee the equality constraint, it requires μ approaching infinity, which may cause bad numerical conditions. Instead, once introducing a Lagrangian multiplier, it is no longer requiring $\mu\to\infty$ (Li 2011; Nocedal and Wright 2006). Thus, we rewrite (7) into the standard ALM formulation as:

$$\min_{\mathbf{A},\mathbf{E},\Lambda,\mu} \mathcal{L}_A=\|\mathbf{E}\|_p^p+\gamma\|\mathbf{A}\|_{2,1}+\frac{\mu}{2}\left\|\mathbf{E}-\mathbf{X}+\mathbf{XA}+\frac{\Lambda}{\mu}\right\|_F^2, \tag{8}$$

where Λ consists of $L\times N$ Lagrangian multipliers. In the following, a highly efficient solver will be given.

The updating rule for Λ Similar to the iterative thresholding (IT) in (Wright et al. 2009; Nie et al. 2014), the degree of violations of the $L\times N$ equality constraints are used to update the Lagrangian multiplier:

$$\Lambda\leftarrow\Lambda+\mu\left(\mathbf{E}-\mathbf{X}+\mathbf{XA}\right), \tag{9}$$

where μ is a monotonically increasing parameter over iteration steps. For example, $\mu\leftarrow\rho\mu$, where $1<\rho<2$ is a predefined parameter (Nocedal and Wright 2006).

Efficient solver for $\mathbf{E}$ Removing irrelevant terms with $\mathbf{E}$ from (8), we have

$$\min_{\mathbf{E}} \|\mathbf{E}\|_p^p+\frac{\mu}{2}\|\mathbf{E}-\mathbf{H}\|_F^2\,, \tag{10}$$

where $\mathbf{H}=\mathbf{X}-\mathbf{XA}-\frac{\Lambda}{\mu}\in\mathbb{R}^{L\times N}$. According to the definition of the ℓ_p-norm and the *Frobenius*-norm, (10) could be decoupled into $L\times N$ independent and unconstrained subproblems. The standard form of these subproblems is

$$\min_{y} f(y)=\lambda|y|^p+\frac{1}{2}(y-c)^2\,, \tag{11}$$

where $\lambda=\frac{1}{\mu}$ is a given positive parameter, y is the scalar variable need to deal with, c is a known scalar constant.

Zuo *et al.* (Zuo et al. 2013) has recently proposed a generalized iterative shrinkage algorithm to solve (11). This algorithm is easy to implement and able to achieve more accurate solutions than current methods. Thus, we use it for our problem as:

$$y^*=\max\left(|c|-\tau_p(\lambda),0\right)\cdot S_p(|c|;\lambda)\cdot\operatorname{sign}(c)\,, \tag{12}$$

where

$$\tau_p(\lambda)=\left[2\lambda(1-p)\right]^{\frac{1}{2-p}}+\lambda p\left[2\lambda(1-p)\right]^{\frac{p-1}{2-p}}$$

; $S_p(|c|;\lambda)$ is obtained by solving the following equation:

$$S_p(c;\lambda)-c+\lambda p\left(S_p(c;\lambda)\right)^{p-1}=0,$$

which could be solved efficiently via an iterative algorithm. In this manner, (10) could be sovled extremely fast.

Accelerated solver for $\mathbf{A}$ The main acceleration focuses on the solver of $\mathbf{A}$. Removing irrelevant terms with $\mathbf{A}$ from (8), we have

$$\min_{\mathbf{A}} \|\mathbf{A}\|_{2,1} + \frac{\beta}{2}\mathrm{Tr}\left\{(\mathbf{X}\mathbf{A}-\mathbf{P})^T(\mathbf{X}\mathbf{A}-\mathbf{P})\right\}, \quad (13)$$

where $\beta = \frac{\mu}{\gamma}$ is a nonnegative parameter, $\mathbf{P} = \mathbf{X} - \mathbf{E} - \frac{\Lambda}{\mu} \in \mathbb{R}^{L\times N}$. Since (13) is convex in $\mathbf{A}$, the optimum could be found by differentiating (13) and setting the derivative to zero. This amounts to tackling the following linear system[2]:

$$\mathbf{A} = \beta\left(\mathbf{V} + \beta\mathbf{X}^T\mathbf{X}\right)^{-1}\mathbf{X}^T\mathbf{P}. \quad (14)$$

As $\mathbf{V} + \beta\mathbf{X}^T\mathbf{X} \in \mathbb{R}^{N\times N}$, (14) is mainly a $N \times N$ linear system. Once solved by the Cholesky factorization, the computational complexity is highly up to $O\left(N^3\right)$. This is by no means a good choice for real applications with large N. In the following, an equivalent derivation of (14) will be proposed to significantly save the computational complexity.

Theorem 2. *The $N \times N$ linear system* (14) *is equivalent to the following $L \times L$ linear system:*

$$\mathbf{A} = \beta\left(\mathbf{X}\mathbf{V}^{-1}\right)^T\left[\mathbf{I}_L + \beta\mathbf{X}\left(\mathbf{X}\mathbf{V}^{-1}\right)^T\right]^{-1}\mathbf{P}, \quad (15)$$

where $\mathbf{I}_L$ is a $L \times L$ identity matrix.

Proof. Note that $\mathbf{V}$ is a $N\times N$ diagonal and positive-definite matrix, the exponent of $\mathbf{V}$ is efficient to achieve, i.e. $\mathbf{V}^\alpha = \{V_{nn}^\alpha\}_{n=1}^N, \forall\alpha\in\mathbb{R}$. We have the following equations

$$\begin{aligned}\mathbf{A} &= \beta\left(\mathbf{V} + \beta\mathbf{X}^T\mathbf{X}\right)^{-1}\mathbf{X}^T\mathbf{P}\\ &= \beta\mathbf{V}^{-\frac{1}{2}}\left[\mathbf{V}^{-\frac{1}{2}}\left(\mathbf{V} + \beta\mathbf{X}^T\mathbf{X}\right)\mathbf{V}^{-\frac{1}{2}}\right]^{-1}\mathbf{V}^{-\frac{1}{2}}\mathbf{X}^T\mathbf{P}\\ &= \beta\mathbf{V}^{-\frac{1}{2}}\left(\mathbf{I}_N + \beta\mathbf{Z}^T\mathbf{Z}\right)^{-1}\mathbf{Z}^T\mathbf{P}, \quad (16)\end{aligned}$$

where $\mathbf{Z} = \mathbf{X}\mathbf{V}^{-\frac{1}{2}}$, $\mathbf{I}_N$ is a $N \times N$ identity matrix. The following equation holds for any conditions

$$\left(\mathbf{I}_N + \beta\mathbf{Z}^T\mathbf{Z}\right)\mathbf{Z}^T = \mathbf{Z}^T\left(\mathbf{I}_L + \beta\mathbf{Z}\mathbf{Z}^T\right). \quad (17)$$

Multiplying (17) with $\left(\mathbf{I}_N + \beta\mathbf{Z}^T\mathbf{Z}\right)^{-1}$ on the left and $\left(\mathbf{I}_L + \beta\mathbf{Z}\mathbf{Z}^T\right)^{-1}$ on the right of both sides of the equal-sign, we have the equation as:

$$\mathbf{Z}^T\left(\mathbf{I}_L + \beta\mathbf{Z}\mathbf{Z}^T\right)^{-1} = \left(\mathbf{I}_N + \beta\mathbf{Z}^T\mathbf{Z}\right)^{-1}\mathbf{Z}^T. \quad (18)$$

Therefore, substituting (18) and $\mathbf{Z} = \mathbf{X}\mathbf{V}^{-\frac{1}{2}}$ into (16), we have the simplified updating rule as:

$$\mathbf{A} = \beta\left(\mathbf{X}\mathbf{V}^{-1}\right)^T\left[\mathbf{I}_L + \beta\mathbf{X}\left(\mathbf{X}\mathbf{V}^{-1}\right)^T\right]^{-1}\mathbf{P}. \quad (19)$$

When $N \gg L$, the most complex operation is the matrix multiplications, not the $L \times L$ linear system. □

Corollary 3. *We have two equivalent updating rules* (14) *and* (15) *for the objective* (13)*. If using* (14) *when $N \le L$, and otherwise using* (15) *as shown in Algorithm 1, the computational complexity of solvers for* (13) *is $O\left(N^2L\right)$. Due to $N \gg L$, we have highly reduced the complexity from $O\left(N^4\right)$ to $O\left(N^2L\right)$ compared with Nie's method.*

Algorithm 1 for (13): $\mathbf{A}^* = \mathrm{ARSS}_{\mathbf{A}}\left(\mathbf{X}, \mathbf{V}, \mathbf{P}, \mathbf{I}_L, \beta\right)$

Input: $\mathbf{X}, \mathbf{V}, \mathbf{P}, \mathbf{I}_L, \beta$

1: **if** $N \le L$ **then**
2: update $\mathbf{A}$ via the updating rule (14), that is
3: $\mathbf{A} = \beta\left(\mathbf{V} + \beta\mathbf{X}^T\mathbf{X}\right)^{-1}\mathbf{X}^T\mathbf{P}$.
4: **else if** $N > L$ **then**
5: update $\mathbf{A}$ via the updating rule (15), that is
6: $\mathbf{A} = \mathbf{B}\left(\mathbf{I}_L + \mathbf{X}\mathbf{B}\right)^{-1}\mathbf{P}$, where $\mathbf{B} = \beta\left(\mathbf{X}\mathbf{V}^{-1}\right)^T$.
7: **end if**

Output: $\mathbf{A}$

Algorithm 2 for (5) or (8): $\mathbf{A}^* = \mathrm{ARSS}_{\mathrm{ALM}}\left(\mathbf{X}, \gamma, p\right)$

Input: $\mathbf{X}, \gamma, p$

1: Initialize $\mu>0, 1<\rho<2, \epsilon=10^{-10}$, $\mathbf{A}=\mathbf{I}_N$, $\Lambda=\mathbf{0}$.
2: **repeat**
3: update $\mathbf{E}$ by the updating rule (12).
4: update $\mathbf{V} = [V_{nn}] \in \mathbb{R}^{N\times N}$.
5: $\mathbf{P} = \mathbf{X}-\mathbf{E}-\frac{\Lambda}{\mu}$, $\beta = \frac{\mu}{\gamma}$; $\mathbf{I}_L$ is a $L \times L$ identity matrix.
6: $\mathbf{A} = \mathrm{ARSS}_{\mathbf{A}}\left(\mathbf{X}, \mathbf{V}, \mathbf{P}, \mathbf{I}_L, \beta\right)$ via **Algorithm** 1.
7: update Λ by the updating rule (9), $\mu \leftarrow \rho\mu$.
8: **until** convergence

Output: $\mathbf{A}$

The solver to update $\mathbf{A}$ is given in Algorithm 1. The overall solver for our model (5) is summarized in Algorithm 2.

According to Theorem 2 and Corollary 3, the solver for our model (13) is highly simplified, as feature length is generally much smaller than data size, i.e $L \ll N$. Similarly, Nie's method could be highly accelerated by Theorem 4, obtaining 500+ times speedup, as shown in Fig. 2 and Table 3.

Theorem 4. *Nie's $N \times N$ solver* (20) *(Nie et al. 2013) is equivalent to the following $L \times L$ linear system* (21)

$$\mathbf{a}_n = U_{nn}\left(U_{nn}\mathbf{X}^T\mathbf{X} + \gamma\mathbf{V}\right)^{-1}\mathbf{X}^T\mathbf{x}_n \quad (20)$$

$$= U_{nn}\left(\mathbf{X}\mathbf{V}^{-1}\right)^T\left(U_{nn}\mathbf{X}\left(\mathbf{X}\mathbf{V}^{-1}\right)^T + \gamma\mathbf{I}_L\right)^{-1}\mathbf{x}_n \quad (21)$$

$\forall n \in \{1, 2, \cdots, N\}$, where $\mathbf{I}_L$ is a $L \times L$ identity matrix.

Proof. Based on (20), we have the following equalities:

$$\begin{aligned}\mathbf{a}_n &= U_{nn}\left(U_{nn}\mathbf{X}^T\mathbf{X} + \gamma\mathbf{V}\right)^{-1}\mathbf{X}^T\mathbf{x}_n,\\ &= U_{nn}\mathbf{V}^{-\frac{1}{2}}\left[\mathbf{V}^{-\frac{1}{2}}\left(U_{nn}\mathbf{X}^T\mathbf{X} + \gamma\mathbf{V}\right)\mathbf{V}^{-\frac{1}{2}}\right]^{-1}\mathbf{V}^{-\frac{1}{2}}\mathbf{X}^T\mathbf{x}_n\\ &= U_{nn}\mathbf{V}^{-\frac{1}{2}}\left(U_{nn}\left(\mathbf{X}\mathbf{V}^{-\frac{1}{2}}\right)^T\mathbf{X}\mathbf{V}^{-\frac{1}{2}} + \gamma\mathbf{I}_N\right)^{-1}\left(\mathbf{X}\mathbf{V}^{-\frac{1}{2}}\right)^T\mathbf{x}_n\\ &= U_{nn}\mathbf{V}^{-\frac{1}{2}}\left(\mathbf{X}\mathbf{V}^{-\frac{1}{2}}\right)^T\left(U_{nn}\mathbf{X}\mathbf{V}^{-\frac{1}{2}}\left(\mathbf{X}\mathbf{V}^{-\frac{1}{2}}\right)^T + \gamma\mathbf{I}_L\right)^{-1}\mathbf{x}_n\\ &= U_{nn}\left(\mathbf{X}\mathbf{V}^{-1}\right)^T\left(U_{nn}\mathbf{X}\left(\mathbf{X}\mathbf{V}^{-1}\right)^T + \gamma\mathbf{I}_L\right)^{-1}\mathbf{x}_n.\end{aligned}$$

The derivations are equivalent; their results are equal. □

[2] $\mathbf{V} \in \mathbb{R}^{N\times N}$ is a positive and diagonal matrix with the n^{th} diagonal entry as $V_{nn} = \frac{1}{\sqrt{\|\mathbf{a}^n\|_2^2 + \epsilon}} > 0$, where ϵ is a small value to avoid singular failures (Nie et al. 2013; Zhu et al. 2014).

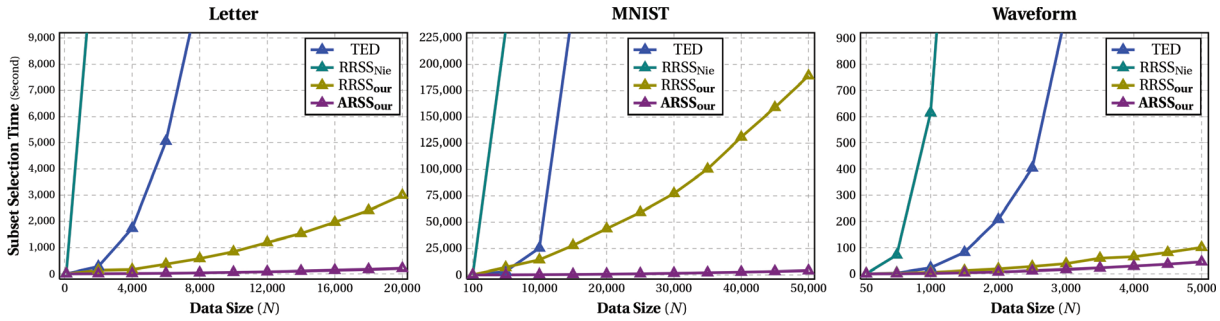

Figure 2: Speed *vs.* increasing N on (a) Letter, (b) MNIST and (c) Waveform. Compared with the authorial solver TED and RRSS_{Nie}, our method ARSS and RRSS_{our} dramatically reduce the computational time. The larger data size is, the larger gaps between these methods are. Note that the selection time is not sensitive to the number of selected samples K. (best viewed in color)

Table 2: Statistics of ten benchmark datasets.

Datasets	Total(N^*)	Candid.(N)	Classes	Features(L)
#1 Vehicle	846	700	4	18
#2 Diabetes	768	600	2	8
#3 Optdigit	5,620	3,823	10	64
#4 Waveform	5,000	4,200	3	21
#5 Satimage	6,435	4,435	7	36
#6 Coil20	1,440	1,200	20	256
#7 University	42,776	17,400	9	115
#8 Center	103,539	4,500	9	115
#9 MNIST	60,000	5,000	10	196
#10 Letter	20,000	13,000	26	16

Corollary 5. *Since feature length is generally much smaller than data size, i.e.* $L \ll N$*, our accelerated solver* (20) *for Nie's model* (3) *is highly faster than the authorial solver* (21)*. Theoretically, we reduce the computational complexity from* $O\left(N^4\right)$ *to* $O\left(N^2L + NL^3\right)$*, while maintaining the same solution. That is, like Nie's solver* (20)*, our speedup solver* (21) *can reach the global optimum. Extensive empirical results will verify the huge acceleration*

Experiments

Experimental Settings

In this part, the experimental settings are introduced. All experiments are conducted on a server with 64-core Intel Xeon E7-4820 @ 2.00 GHz, 18 Mb Cache and 0.986 TB RAM, using Matlab 2012. Brief descriptions of ten benchmark datasets are summarized in Table 2, where 'Total(N^*)' denotes the total set of samples in each data. Due to the high computational complexity, other methods can only handle small datasets (while our method can handle the total set). Thus, we randomly choose the candidate set from the total set to reduce the sample size, i.e. $N < N^*$ (cf. 'Total(N^*)' and 'candid.(N)' in Table 2). The remainder (except candidate set) are used for test. Specifically, to simulate the varying quality of samples, ten percentage of candidate samples from each class are randomly selected and arbitrarily added one of the following three kinds of noise: "Gaussian", "Laplace" and "Salt & pepper" respectively. In a word, all experiment settings are same and fair for all the methods.

Speed Comparisons

There are two parts of speed comparisons. First, how speed varies with increasing N is illustrated in Fig. 2. Then the comparison of specific speed is summarized in Table 3. Note that TED and RRSS_{Nie} denote the authorial solver (via authorial codes); RRSS_{our} is our accelerated solver for Nie's model via Theorem 4; ARSS is the proposed method.

Speed *vs.* increasing N To verify the great superiority of our method over the state-of-the-art methods in speed, three experiments are conducted. The results are illustrated in Fig. 2, where there are three sub-figures showing the speed of four methods on the benchmark datasets of Letter, MNIST and Waveform respectively. As we shall see, both selection time of TED (Yu, Bi, and Tresp 2006) and RRSS_{Nie} (Nie et al. 2013) increases dramatically as N increases. No surprisingly, RRSS_{Nie} is incredibly time-consuming as N grows—the order of curves looks higher than quadratic. Actually, the theoretical complexity of RRSS_{Nie} is highly up to $O\left(N^4\right)$ as analyzed in Remark 1.

Compared with TED and RRSS_{Nie}, the curve of ARSS is surprisingly lower and highly stable against increasing N; there is almost no rise of selection time over growing N. This is owing to the speedup techniques of ALM and equivalent derivations. Via them, we reduce the computational cost from $O\left(N^4\right)$ to $O\left(N^2L\right)$, as analyzed in Theorem 2 and Corollary 3. Moreover, with the help of Theorem 4, RRSS_{our} is the second faster algorithm that is significantly accelerated compared with the authorial algorithm RRSS_{Nie}.

Speed with fixed N The speed of four algorithms is summarized in Table 3a, where each row shows the results on one dataset and the last row displays the average results. Four conclusions can be drawn from Table 3a. First, ARSS is the fastest algorithm, and RRSS_{our} is the second fastest algorithm. Second, with the help of Theorem 4, RRSS_{our} is highly faster than RRSS_{Nie}, averagely obtaining a 559 times acceleration. Third, ARSS is dramatically faster than RRSS_{Nie} and TED; the results in Table 3a verify an average acceleration of 23275 times faster than RRSS_{Nie} and 281 times faster than TED. This means that for example if it

Table 3: Performances of TED, RRSS and ARSS: (***left-a***) speed in seconds, (***right-b***) prediction accuracies. In terms of speed, with the help of Theorem 4, RRSS_{our} is averagely 559+ times faster than the authorial algorithm, i.e. RRSS_{Nie}; ARSS achieves surprisingly 23275+ times acceleration compared with RRSS_{Nie}. Due to the more robust loss in the ℓ_p-norm, the prediction accuracy of ARSS is highly encouraging.

Datasets	Speed Comparison in seconds					Accuracy by KNN (%)			Accuracy by Linear SVM (%)		
	TED	RRSS_{Nie}	RRSS_{our}	**ARSS**	ARSS(N^*)	TED	RRSS	**ARSS**	TED	RRSS	**ARSS**
Vehicle	33.1	1399.4	*28.2*	**2.4**	3.6	58.904	**63.014**	62.329	71.233	75.342	**76.027**
Diabetes	15.4	1271.3	*2.1*	**0.8**	2.5	58.929	68.452	**69.643**	66.667	73.810	**74.405**
Optdigit	3082.6	138486.8	*637.7*	**36.2**	46.9	89.816	91.208	**91.375**	91.931	91.430	**92.265**
Waveform	3467.5	147095.2	*597.7*	**12.9**	28.2	73.875	79.375	**80.375**	83.500	**85.500**	84.875
Satimage	2651.8	115158.5	*416.3*	**21.4**	45.0	**82.000**	69.950	79.500	78.300	67.450	**79.100**
University	83206.0	>9624600.0	*23409.2*	**507.3**	1173.3	64.131	**70.941**	69.684	66.776	71.635	**72.679**
Center	9742.3	581159.9	*2366.5*	**48.3**	6539.9	30.864	51.954	**53.845**	42.872	59.899	**61.907**
MNIST	8785.8	1092390.0	*5558.8*	**86.6**	2944.3	75.545	**78.667**	76.985	82.611	**83.684**	82.116
Letter	120577.9	7515330.0	*1351.9*	**109.8**	157.0	29.843	40.929	**41.543**	37.843	46.886	**47.157**
Average	25729.1	>2135210.1	*3818.7*	**91.7**	1215.6	62.656	68.277	**69.475**	69.081	72.848	**74.503**

*'ARSS(N^*)' means the task of selecting samples from the whole dataset (with N^* samples as shown in the 2nd column in Table 2), while 'TED' to '**ARSS**' indicate the problem of dealing with the candidate sample sets (with N samples as shown in the 3rd column in Table 2).*

takes RRSS_{Nie} 100 years to do a subset selection task, it only takes our method 1.6 days to address the same problem. Finally, we apply ARSS to the whole sample set of each data. The results are displayed in the 6$^{\text{th}}$ column in Table 3, showing its capability to process very large datasets.

Prediction Accuracy

Accuracy comparison We conduct experiments on ten benchmark datasets. For each dataset, the top 200 representative samples are selected for training. The prediction accuracies are reported in Table 3b, including the results of two popular classifiers. Three observations can be drawn from this table. First, Linear SVM generally outperforms KNN. Second, in general, our method performs the best; for a few cases, our method achieves comparable results with the best performances. Third, compared with TED, both RRSS and ARSS achieve an appreciable advantage. The above analyses are better illustrated in the last row of Table 3b. These results demonstrate that the ℓ_p loss in our model is well suited to select exemplars from the sample sets of various quality.

Prediction accuracies *vs.* increasing K To give a more detailed comparison, Fig. 3 shows the prediction accuracies versus growing K (the number of selected samples). There are two rows and four columns of sub-figures. The top row shows the results of KNN, and the bottom one shows results of SVM. Each column gives the result on one dataset. As we shall see, the prediction accuracies generally increase as K increases. Such case is consistent with the common view that more training data will boost the prediction accuracy. For each sub-figure, ARSS is generally among the best. This case implies that our robust objective (5) via the ℓ_p-norm is feasible to select subsets from the data of varying qualities.

Conclusion

To deal with tremendous data of varying quality, we propose an accelerated robust subset selection (ARSS) method. The ℓ_p-norm is exploited to enhance the robustness against both outlier samples and outlier features. Although the resulted objective is complex to solve, we propose a highly efficient solver via two techniques: ALM and equivalent derivations. Via them, we greatly reduce the computational complexity from $O\left(N^4\right)$ to $O\left(N^2L\right)$. Here feature length L is much smaller than data size N, i.e. $L \ll N$. Extensive results on ten benchmark datasets verify that our method not only runs 10,000+ times faster than the most related method, but also outperforms state of the art methods. Moreover, we propose an accelerated solver to highly speed up Nie's method, theoretically reducing the computational complexity from $O\left(N^4\right)$ to $O\left(N^2L + NL^3\right)$. Empirically, our accelerated solver could achieve equal results and 500+ times acceleration compared with the authorial solver.

Limitation. Our efficient algorithm build on the observation that the number of samples is generally larger than feature length, i.e. $N > L$. For the case of $N \leq L$, the acceleration will be inapparent.

Acknowledgements

The authors would like to thank the editor and the reviewers for their valuable suggestions. Besides, this work is supported by the projects (Grant No. 61272331, 91338202, 61305049 and 61203277) of the National Natural Science Foundation of China.

References

Bien, J., and Tibshirani, R. 2011. Prototype selection for interpretable classification. *Annals of Applied Statistics* 5(4):2403–2424.

Boutsidis, C.; Mahoney, M. W.; and Drineas, P. 2009. An improved approximation algorithm for the column subset selection problem. In *SODA*, 968–977.

Chan, T. F. 1987. Rank revealing {QR} factorizations. *Linear Algebra and its Applications* 88–89(0):67 – 82.

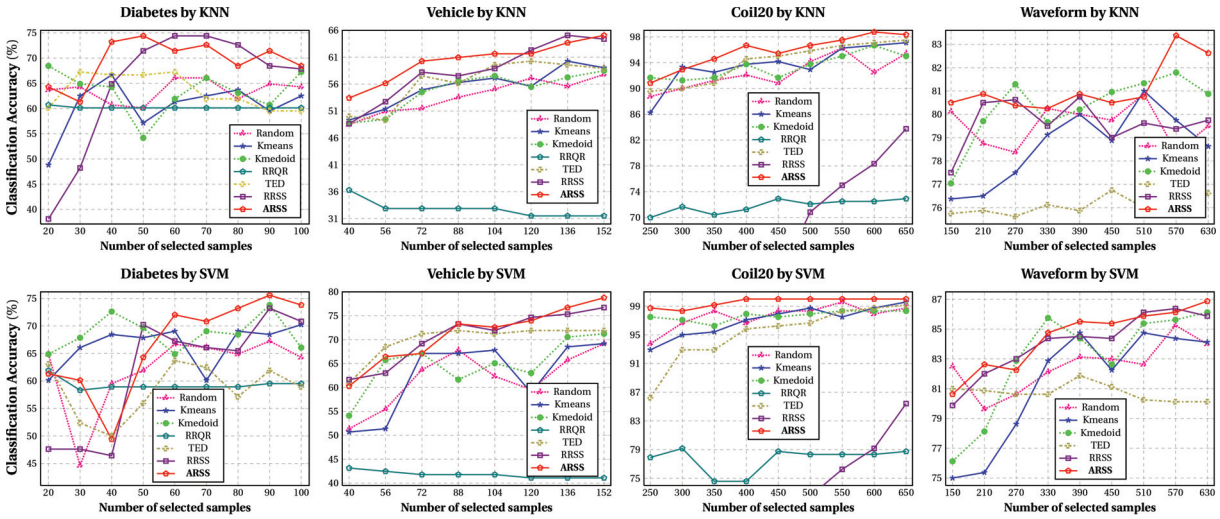

Figure 3: Accuracies *vs.* increasing K (the number of selected samples). There are two rows and four columns of subfigures: the top row shows the prediction accuracies of KNN, and the bottom shows the results of Linear SVM; each column shows the performances on one datasets, that is Diabetes, Vehicle, Coil20 and Waveform respectively. Generally, ARSS (ours) is among the best. (best viewed in color)

Elhamifar, E.; Sapiro, G.; Vidal, R.; and Vidal, R. 2012a. See all by looking at a few: Sparse modeling for finding representative objects. In *IEEE CVPR*, 1600–1607.

Elhamifar, E.; Sapiro, G.; Vidal, R.; and Vidal, R. 2012b. Finding exemplars from pairwise dissimilarities via simultaneous sparse recovery. In *NIPS*, 19–27.

Frey, B. J., and Dueck, D. 2007. Clustering by passing messages between data points. *Science* 315(5814):972–976.

Garcia, S.; Derrac, J.; Cano, J. R.; and Herrera, F. 2012. Prototype selection for nearest neighbor classification: Taxonomy and empirical study. *IEEE Trans. Pattern Anal. Mach. Intell.* 34(3):417–435.

Jenatton, R.; Audibert, J.-Y.; and Bach, F. 2011. Structured variable selection with sparsity-inducing norms. *Journal of Machine Learning Research (JMLR)* 12:2777–2824.

Levin, A.; Lischinski, D.; Weiss, Y.; and Weiss, Y. 2008. A closed-form solution to natural image matting. *IEEE Trans. Pattern Anal. Mach. Intell.* 30(2):228–242.

Li, C. 2011. *Compressive Sensing for 3D Data Processing Tasks: Applications, Models and Algorithms*. Ph.D. Dissertation, Houston, TX, USA. AAI3524544.

Liu, G.; Lin, Z.; Yu, Y.; and Yu, Y. 2010. Robust subspace segmentation by low-rank representation. In *ICML*, 663–670.

Meng, G.; Wang, Y.; Duan, J.; Xiang, S.; and Pan, C. 2013. Efficient image dehazing with boundary constraint and contextual regularization. In *ICCV*, 617–624.

Nie, F.; Wang, H.; Cai, X.; Huang, H.; and Ding, C. 2012. Robust matrix completion via joint schatten p-norm and lp-norm minimization. In *IEEE ICDM*, 566–574.

Nie, F.; Wang, H.; Huang, H.; and Ding, C. H. Q. 2013. Early active learning via robust representation and structured sparsity. In *IJCAI*, 1572–1578.

Nie, F.; Huang, Y.; Wang, X.; and Huang, H. 2014. New primal svm solver with linear computational cost for big data classifications. In *ICML*.

Nocedal, J., and Wright, S. J. 2006. *Numerical Optimization*. New York: Springer, 2nd edition.

Wang, H.; Nie, F.; Huang, H.; and Huang, H. 2014. Robust distance metric learning via simultaneous l1-norm minimization and maximization. In *ICML*, 1836–1844.

Wang, J.; Kumar, S.; and Chang, S.-F. 2012. Semi-supervised hashing for large-scale search. *IEEE Trans. Pattern Anal. Mach. Intell.* 34.

Wright, J.; Ganesh, A.; Rao, S.; Peng, Y.; and Ma, Y. 2009. Robust principal component analysis: Exact recovery of corrupted low-rank matrices via convex optimization. In *NIPS*, 2080–2088.

Yu, K.; Zhu, S.; Xu, W.; ; and Gong, Y. 2008. Non-greedy active learning for text categorization using convex transductive experimental design. In *SIGIR*, 1081–1088.

Yu, K.; Bi, J.; and Tresp, V. 2006. Active learning via transductive experimental design. In *ICML*, 1081–1088.

Zhu, F.; Wang, Y.; Fan, B.; Meng, G.; and Pan, C. 2014. Effective spectral unmixing via robust representation and learning-based sparsity. *arXiv* :1409.0685.

Zuo, W.; Meng, D.; Zhang, L.; Feng, X.; and Zhang, D. 2013. A generalized iterated shrinkage algorithm for non-convex sparse coding. In *IEEE ICCV*, 217–224.