Self-Supervised Deep Low-Rank Assignment Model for Prototype Selection

Xingxing Zhang, Zhenfeng Zhu, Yao Zhao, Deqiang Kong
Institute of Information Science, Beijing Jiaotong University, Beijing, China
Beijing Key Laboratory of Advanced Information Science and Network Technology, Beijing, China
{zhangxing,zhfzhu,yzhao,dqkong}@bjtu.edu.cn

Abstract

Prototype selection is a promising technique for removing redundancy and irrelevance from large-scale data. Here, we consider it as a task assignment problem, which refers to assigning each element of a source set to one representative, i.e., prototype. However, due to the outliers and uncertain distribution on source, the selected prototypes are generally less representative and interesting. To alleviate this issue, we develop in this paper a Self-supervised Deep Low-rank Assignment model (SDLA). By dynamically integrating a low-rank assignment model with deep representation learning, our model effectively ensures the goodness-of-exemplar and goodness-of-discrimination of selected prototypes. Specifically, on the basis of a denoising autoencoder, dissimilarity metrics on source are continuously self-refined in embedding space with weak supervision from selected prototypes, thus preserving categorical similarity. Conversely, working on this metric space, similar samples tend to select the same prototypes by designing a low-rank assignment model. Experimental results on applications like text clustering and image classification (using prototypes) demonstrate our method is considerably superior to the state-of-the-art methods in prototype selection.

1 Introduction

Prototype selection is the task of finding exemplar samples, called prototypes, from a large collection of data points. This is at the center of many problems in data analysis and processing field because it holds several advantages over data storage, compression, synthesis, cleansing and visualization. First, the memory cost for storing information on the data can be significantly reduced using prototypes. Second, prototypes help in clustering of data, and, as the most prototypical samples, can be used for efficient synthesis of new data points. More importantly, the computational efficiency for data modeling, such as classifier training [Garcia *et al.*, 2012; Zhang *et al.*, 2018] and active learning [Lin *et al.*, 2018], can be improved by working on prototypes. In addition, selecting prototypes helps to remove redundant or irrelevant points, such as outliers. Finally, prototype selection has been applied for anomaly detection [Cong *et al.*, 2011], web media summarization and recommendation [Meng *et al.*, 2016; Cong *et al.*, 2017] (see Figure 1), segmentation of dynamic data [Elhamifar *et al.*, 2016], and more.

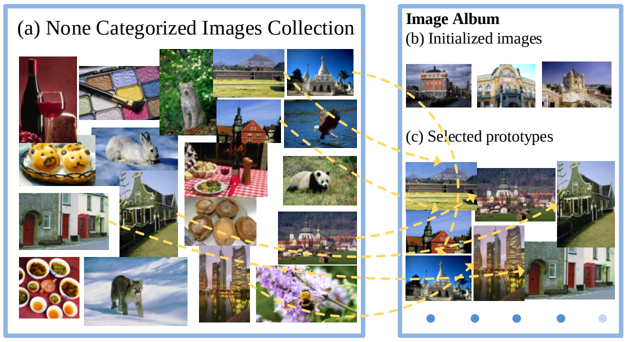


Figure 1: Assisted photo albuming (i.e. web media recommendation) by prototype selection. When we want to build a new photo album, the only thing we need is to select exemplar samples related to the initialized topic.

To characterize the informativeness of prototypes in terms of ability to represent the entire distribution, many selection strategies have been proposed, generally including two categories. The first one involves greedy methods that select next prototype with respect to the previously selected prototypes, and usually maximizes submodular functions, such as graph-cuts and facility location [Wang and Zhang, 2013; Elhamifar and Kaluza, 2017; Krause *et al.*, 2008]. Obviously, they usually result in high computational complexity. The second category is comprised of model-driven methods that aim to maximize global objective based on subspace leaning or pairwise relationship. Their resulting solutions are closely related to a variety of optimization algorithms.

Concretely, subspace leaning based models mainly focus on the data that lie in one or several low-dimensional subspaces. The Rank Revealing QR algorithm [Chan, 1987] is a representative one. By representing data in a low dimensional space with possibly minimal representation error, various effective models have been formulated to learn a selection matrix, such as the sparse dictionary selection (SDS) method [Cong *et al.*, 2011] and sparse modeling representative selection (SMRS) model [Elhamifar *et al.*, 2012].

To further exploit data structure, some improved methods have been proposed [Dornaika and Aldine, 2015; Wang *et al.*, 2017]. For instance, structured sparse dictionary selection (SSDS) [Wang *et al.*, 2017] method tried to select prototypes with both representativeness and diversity via three structured regularizers. However, by multi-linear coding criteria, these methods are effective only when the samples from different groups are sufficiently dissimilar.

In contrast, pairwise relationship based models aim at the data that could be naturally grouped under a certain dissimilarity metric. One naive approach is Kmedoids [Kaufman and Rousseeuw, 1987], which finds k medoid centers as prototypes by pursuing the minimum total distance from all samples. Some variants [Nellore and Ward, 2015] were further proposed based on Kmedoids. Unlike Kmeans [Duda *et al.*, 2012] or Kmedoids, Affinity Propagation (AP) algorithm [Frey and Dueck, 2007] does not require any initialization for prototypes, but has suboptimal property. To tackle this issue, dissimilarity-based sparse subset selection (DS3) method [Elhamifar *et al.*, 2016] was recently proposed to select prototypes via a trace minimization model. In general, this kind of methods performs well only under appropriate similarity metric.

Furthermore, some real datasets do not live in a vector space, e.g., social network data or proteomics data. Therefore, model-driven methods based on pairwise relationship have more advantages on prototype selection. However, most existing algorithms suffer from imposing restrictions on the type of pairwise relationship. More importantly, unreliable metric caused by outliers or uncertain data distribution significantly reduces the quality of prototypes. Inspired by representation learning [Song *et al.*, 2017], we consider introducing a unified framework for dissimilarity learning and prototype selection. As a result, the metric is efficiently self-refined with weak supervision from prototypes, and conversely, prototypes are more informative by using refined metrics. These construct the basic *motivation* of our framework.

In summary, the main *contributions* of this work are highlighted as follows.

- By considering the prototype selection as a task assignment problem, we develop a Self-supervised Deep Low-rank Assignment model (SDLA), which aims to jointly learn ideal dissimilarity metrics in embedding space and select informative prototypes in metric space.
- Unlike DS3 [Elhamifar *et al.*, 2016], inspired by the good performance of deep representation learning, we propose to learn dissimilarity metrics based on a variant of a denoising autoencoder, in which categorical similarity is preserved with weak supervision from prototypes.
- Working on the learned metric space, similar samples tend to select the same prototypes by designing a low-rank assignment model, thus guaranteeing a diversified selection.
- The quality of prototypes selected by the proposed framework is reasonably evaluated with examples in text clustering and image classification (using prototypes), showing very promising results.

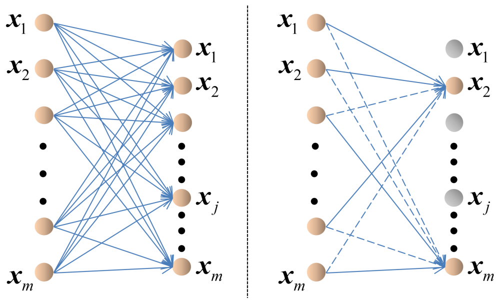


Figure 2: Illustration of prototype selection. Left: The full connection of m samples. Right: The entire source set is assigned to an optimal subset of it, called prototype set.

2 The Proposed Framework

2.1 Problem Statement

Let $\boldsymbol{X} = [\boldsymbol{x}_1, \ldots, \boldsymbol{x}_m] \in \mathbb{R}^{d\times m}$ be the source data matrix of m samples in $\mathbb{R}^d$. As shown in Figure 2, we consider the problem of selecting an interesting subset of m samples, called prototype set, that can efficiently describe and summarize all samples in source data. Inspired by DS3 [Elhamifar *et al.*, 2016], we primarily pursue the minimum assignment cost based on pairwise relationships on source.

2.2 Problem Formulation

It is noteworthy that pairwise dissimilarity matrix $\boldsymbol{D} = \{d_{ij}\}_{i=1,\ldots,m}^{j=1,\ldots,m}$ between samples is directly given as the input to DS3 [Elhamifar *et al.*, 2016]. Actually, the given $\{d_{ij}\}$, such as Euclidean distance or χ^2 distance, may be not discriminative due to uncertain distribution and outliers in $\boldsymbol{X}$, thus leading to a less representative and diversified selection. In light of the good performance of representation learning [Bengio *et al.*, 2013], we consider circularly refining the dissimilarity metric and the selection indicator. Thereby, available semantic information from selected prototypes can be employed to learn the parameters of metric architecture, while conversely, more discriminative metric can improve the quality of prototypes. Towards this end, we develop a Self-supervised Deep Low-rank Assignment model (SDLA) for prototype selection as follows:

$$\begin{aligned}
\min_{\Theta,\{z_{ij}\}} &\sum_{j=1}^{m}\sum_{i=1}^{m} \Phi_{\mathrm{dSim}}(\Phi_{\mathrm{rep}}(\Theta;\boldsymbol{x}_i),\Phi_{\mathrm{rep}}(\Theta;\boldsymbol{x}_j))z_{ij}\\
&+\omega_1\Omega(\Theta)+\omega_2\Psi(\boldsymbol{Z}) \qquad (1)\\
s.t. \quad &\sum_{i=1}^{m} z_{ij}=1, \forall j; \quad z_{ij}\geq 0, \forall i,j,
\end{aligned}$$

where $\Phi_{\mathrm{rep}}(\Theta;\cdot)$ learns a deep representation of a sample, and Θ is the set of related parameters. $\Phi_{\mathrm{dSim}}(\cdot,\cdot)$ aims to capture the dissimilarity of two samples in learned embedding space, thus $d_{ij} = \Phi_{\mathrm{dSim}}(\Phi_{\mathrm{rep}}(\Theta;\boldsymbol{x}_i),\Phi_{\mathrm{rep}}(\Theta;\boldsymbol{x}_j))$. $\boldsymbol{Z} = \{z_{ij}\}_{i=1,\ldots,m}^{j=1,\ldots,m}$ is a selection matrix, $z_{ij} \in [0,1]$ is the probability of selecting $\boldsymbol{x}_i$ as the prototype of $\boldsymbol{x}_j$, thus enforcing $\sum_{i=1}^{m} z_{ij} = 1$ for $\boldsymbol{x}_j$. The first term in the objective function corresponds to the total assignment cost, the second term corresponds to the constraints in the deep architecture,

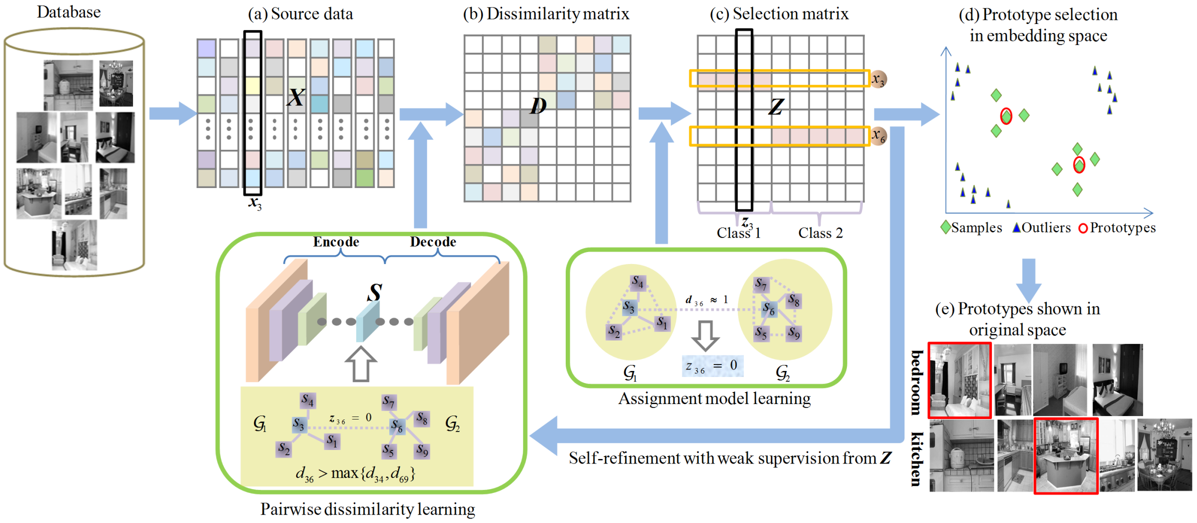

Database
(a) Source data
X
x3
(b) Dissimilarity matrix
D
(c) Selection matrix
Z
x3
x6
z3
Class 1
Class 2
(d) Prototype selection in embedding space
Samples
Outliers
Prototypes
(e) Prototypes shown in original space
bedroom
kitchen
Encode
Decode
S
s1 s2 s3 s4 s5 s6 s7 s8 s9
G1
G2
z36 = 0
d36 > max{d34, d69}
Pairwise dissimilarity learning
d36 ≈ 1
z36 = 0
Assignment model learning
Self-refinement with weak supervision from Z

Figure 3: Framework of prototype selection. First, we convert multimedia into source data. Then, a cyclic self-refinement learning way between dissimilarity learning in embedding space and assignment model learning for prototype selection is implemented. Finally, we find the optimal selection indicator, by which we select the discriminative prototypes and obtain a partitioning simultaneously.

and the last term corresponds to the number of prototypes. ω_1 and ω_2 are nonnegative parameters to balance these terms. Figure 3 provides an illustration of the proposed model in (1).

Concretely, for any $\boldsymbol{x}_i$ and $\boldsymbol{x}_j$, if $d_{ij} = 0$, then $z_{ij} = 1$ due to low assignment cost, which implies that $\boldsymbol{x}_i$ and $\boldsymbol{x}_j$ are from the same class. While $z_{ij} = 0$ if $d_{ij} = \infty$, which implies that $\boldsymbol{x}_i$ and $\boldsymbol{x}_j$ are from different classes. Since $\boldsymbol{Z}$ provides additional semantic information for all samples, $\boldsymbol{D}$ will be learned more discriminatively by preserving categorical similarity with weak supervision from $\boldsymbol{Z}$.

Considering the separability of both objective and constraints in (1), we further divide (1) into two subproblems $\mathbb{P}_1$ and $\mathbb{P}_2$, which are cyclic self-refinement:

- $\mathbb{P}_1$ corresponding to the **pairwise dissimilarity learning** module in Figure 3, is written as

$$\min_{\Theta} \sum_{i,j=1}^{m} \Phi_{\text{dSim}}(\Phi_{\text{rep}}(\Theta;\boldsymbol{x}_i), \Phi_{\text{rep}}(\Theta;\boldsymbol{x}_j))z_{ij} + \omega_1\Omega(\Theta) \tag{2}$$

- $\mathbb{P}_2$ corresponding to the **assignment model learning** module in Figure 3, is written as

$$\min_{\{z_{ij}\}} \sum_{i,j=1}^{m} \Phi_{\text{dSim}}(\Phi_{\text{rep}}(\Theta;\boldsymbol{x}_i), \Phi_{\text{rep}}(\Theta;\boldsymbol{x}_j))z_{ij} + \omega_2\Psi(\boldsymbol{Z})$$
$$s.t. \quad \sum_{i=1}^{m} z_{ij} = 1, \forall j; \quad z_{ij} \geq 0, \forall i,j, \tag{3}$$

The details about learning Θ and $\boldsymbol{Z}$ are introduced in Section 3.1 and Section 3.2, respectively. As a result, the indices of nonzero rows of the solution $\boldsymbol{Z}^*$ correspond to the indices of those samples that are chosen as the data prototypes. In addition, $\boldsymbol{Z}^*$ shows the membership of samples in $\boldsymbol{X}$ to prototypes. That is, $\boldsymbol{z}_j^* = \left[z_{1j}^*, \ldots, z_{mj}^*\right]^{\mathrm{T}} \in \mathbb{R}^m$ corresponds to the probability vector of $\boldsymbol{x}_j$ being represented by each sample in $\boldsymbol{X}$. Therefore, we can obtain a partitioning of $\boldsymbol{X}$ under the rule that, if $\chi = \{\boldsymbol{x}_{l_1}, \ldots, \boldsymbol{x}_{l_\kappa}\}$ denotes the set of selected prototypes, we can assign $\boldsymbol{x}_j$ to the prototype $\boldsymbol{x}_{\delta_j}$ by

$$\delta_j = \arg \min_{i\in\{l_1,\ldots,l_\kappa\}} z_{ij}^* \tag{4}$$

Consequently, $\boldsymbol{X}$ is classified into κ groups corresponding to κ prototypes via the selection matrix $\boldsymbol{Z}^*$.

3 Optimization

3.1 Pairwise Dissimilarity Learning

As shown in (2), optimizing Θ aims to refine pairwise dissimilarities $\{d_{ij}\}$, and it seamlessly connects the visual content and dissimilarity metric with weak supervision from $\boldsymbol{Z}$. However, due to the outliers and uncertain distribution on $\boldsymbol{X}$, we consider generating distributed representation vectors and pairwise dissimilarities on the basis of a denoising autoencoder [Vincent *et al.*, 2008]. The general denoising autoencoder is formulated as:

$$\begin{aligned}
&\tilde{\boldsymbol{x}}_i \sim q(\tilde{\boldsymbol{x}}_i|\boldsymbol{x}_i);\\
&\boldsymbol{s}_i = f(\boldsymbol{W}\tilde{\boldsymbol{x}}_i + \boldsymbol{b});\\
&\boldsymbol{y}_i = f(\boldsymbol{W}'\boldsymbol{s}_i + \boldsymbol{b}');\\
&L_{\text{R}}(\boldsymbol{y}_i, \boldsymbol{x}_i) = \|\boldsymbol{x}_i - \boldsymbol{y}_i\|_2\,;\\
&\Theta = \arg \min_{\boldsymbol{W},\boldsymbol{W}',\boldsymbol{b},\boldsymbol{b}'} \sum_{i=1}^{m} L_{\text{R}}(\boldsymbol{y}_i, \boldsymbol{x}_i).
\end{aligned} \tag{5}$$

where $\boldsymbol{x}_i \in \boldsymbol{X}$ is the original input vector, $i = 1, \ldots, m$, and $q(\cdot|\cdot)$ is the corrupting distribution. The stochastically corrupted vector, $\tilde{\boldsymbol{x}}_i$, is obtained from $q(\cdot|\boldsymbol{x}_i)$. Generally, $\tilde{\boldsymbol{x}}_i = (1-\beta)\boldsymbol{x}_i$, and β is the corruption rate in the training phase. The hidden representation, $\boldsymbol{s}_i$, is mapped from $\tilde{\boldsymbol{x}}_i$ through the network, which consists of an activation function $f(\cdot)$, parameter matrix $\boldsymbol{W}$, and parameter vector $\boldsymbol{b}$. In the

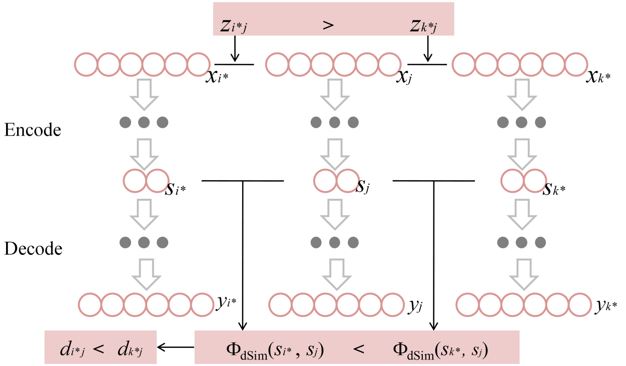


Figure 4: Encoder for triplets of samples, in which d_{i^*j} is smaller than d_{k^*j} if z_{i^*j} is much larger than z_{k^*j}.

same way, the reconstructed vector, $\boldsymbol{y}_i$, is also mapped from $\boldsymbol{s}_i$ with parameter matrix $\boldsymbol{W}'$ and parameter vector $\boldsymbol{b}'$. Using a loss function, $L_R(\cdot,\cdot)$, we learn these parameters to minimize the total reconstruction error of $\{\boldsymbol{y}_i\}$ and $\{\boldsymbol{x}_i\}$.

Here, our proposed deep framework contains L layers of nonlinear transformation, and the output $\boldsymbol{s}_i$ of the middle layer is usually used as a representation vector that corresponds to $\boldsymbol{x}_i$ [Okura *et al.*, 2017], thus $\boldsymbol{s}_i = \Phi_{\mathrm{rep}}(\Theta; \boldsymbol{x}_i)$. Then d_{ij} is captured in embedding space. In this work, we define $d_{ij} = \Phi_{\mathrm{dSim}}(\boldsymbol{s}_i, \boldsymbol{s}_j) = 1 - \boldsymbol{s}_i^{\mathrm{T}} \boldsymbol{s}_j$. However, $\boldsymbol{s}_i$ in (5) only holds the information of $\boldsymbol{x}_i$. As shown in Figure 4, it is expected that the distance between two representation vectors $\boldsymbol{s}_i$ and $\boldsymbol{s}_j$, i.e., d_{ij}, is smaller, if $\boldsymbol{x}_j$ is more similar to $\boldsymbol{x}_i$ compared with $\boldsymbol{x}_k$ (which is evaluated by $z_{ij} > z_{kj}$). This is just the **weak supervision** from selected prototypes via $\boldsymbol{Z}$. For this end, we add a triplet loss $L_{\mathcal{T}}$ to the objective as follows:

$$L_{\mathcal{T}}(\boldsymbol{x}_i, \boldsymbol{x}_j, \boldsymbol{x}_k) = \log(1 + \exp(d_{ij} - d_{kj})); \tag{6}$$

In essence, the loss function $L_{\mathcal{T}}$ is a penalty function for sample similarity to correspond to categorical similarity.

In addition, as observed in (2), the assignment cost $L_{\mathcal{E}}$ should be as small as possible.

$$L_{\mathcal{E}}(\boldsymbol{x}_i, \boldsymbol{x}_j) = d_{ij} z_{ij}; \tag{7}$$

Consequently, the total loss function $L_A(\boldsymbol{X})$ is rewritten as follows:

$$\begin{aligned} L_A(\boldsymbol{X}) = & \underbrace{\sum_{i=1}^{m} L_{\mathrm{R}}(\boldsymbol{y}_i, \boldsymbol{x}_i)}_{\textbf{Reconstruction loss}} + \alpha_1 \underbrace{\sum_{j=1}^{m} \sum_{i=1}^{m} L_{\mathcal{E}}(\boldsymbol{x}_i, \boldsymbol{x}_j)}_{\textbf{Assignment loss in (2)}} \\ & + \alpha_2 \underbrace{\sum_{(i,j,k)\in\mathcal{T}} L_{\mathcal{T}}(\boldsymbol{x}_i, \boldsymbol{x}_j, \boldsymbol{x}_k)}_{\textbf{Self-supervised triplet loss}}; \end{aligned} \tag{8}$$

where α_1 and α_2 are two hyperparameters to balance three loss terms. $\mathcal{T}$ is the set of triplets, which is constructed by associating each positive pair in the minibatch with a semi-hard negative sample. Specifically, $\boldsymbol{x}_{i^*}$ and $\boldsymbol{x}_{k^*}$ denote the positive and negative samples of $\boldsymbol{x}_j$ in a triplet, and

$$\begin{aligned} i^* &= \arg \max_{i\in\{1,\ldots,m\}} z_{ij}; \\ k^* &= \arg \min_{k\in\{1,\ldots,m\}} z_{kj}. \end{aligned} \tag{9}$$

Finally, we use the elementwise sigmoid function as $f(\cdot)$, and masking noise as $q(\cdot|\cdot)$. The following model is trained by mini-batch Adaptive Moment Estimation:

$$\Theta = \arg \min_{\{\boldsymbol{W}_l, \boldsymbol{W}_l{}', \boldsymbol{b}_l, \boldsymbol{b}_l{}'\}} L_A(\boldsymbol{X}) + \omega_1 \Omega(\Theta) \tag{10}$$

where $\Omega(\Theta) = \sum_{l=1}^{L/2} (\|\boldsymbol{W}_l\|_F^2 + \|\boldsymbol{b}_l\|_2^2 + \|\boldsymbol{W}_l{}'\|_F^2 + \|\boldsymbol{b}_l{}'\|_2^2)$, and $l = 1, \ldots, L/2$.

3.2 Assignment Model Learning

As shown in (3), optimizing $\boldsymbol{Z}$ aims to select a small number of prototypes with both diversity and representativeness. Then we consider enforcing the lowest rank and sparsity properties on $\boldsymbol{Z}$. That is, $\Psi(\boldsymbol{Z}) = \lambda_1 \mathrm{rank}(\boldsymbol{Z}) + \lambda_2 \mathrm{card}(\boldsymbol{Z})$, where $\lambda_1, \lambda_2 > 0$, $\mathrm{rank}(\boldsymbol{Z})$ denotes the rank of $\boldsymbol{Z}$, and $\mathrm{card}(\boldsymbol{Z})$ denotes the number of nonzero elements of $\boldsymbol{Z}$. As observed in [Zhuang *et al.*, 2012], the low-rankness criterion is better at capturing the global structure of dissimilarity $\boldsymbol{D}$, while the sparsity criterion can capture the local structure of each data vector. Specifically, low-rankness can encourage similar samples to have similar codes (i.e., i^{th} and j^{th} columns of $\boldsymbol{Z}$), so as to select the same prototypes. Thus, (3) is rewritten as

$$\begin{aligned} & \min_{\boldsymbol{Z}} \mathrm{tr}(\boldsymbol{D}^{\mathrm{T}} \boldsymbol{Z}) + \lambda_1 \|\boldsymbol{Z}\|_* + \lambda_2 \|\boldsymbol{Z}\|_0 \\ & s.t. \quad \boldsymbol{1}^{\mathrm{T}} \boldsymbol{Z} = \boldsymbol{1}^{\mathrm{T}}; \quad \boldsymbol{Z} \geq \boldsymbol{0}, \end{aligned} \tag{11}$$

where $\sum_{i,j=1}^{m} d_{ij} z_{ij} = \mathrm{tr}(\boldsymbol{D}^{\mathrm{T}} \boldsymbol{Z})$, $\mathrm{tr}(\cdot)$ denotes the trace operator, and d_{ij} has been obtained from (2). $\|\cdot\|_*$ and $\|\cdot\|_0$ are the nuclear norm and ℓ_0-norm of a matrix, respectively. λ_1 and λ_2 are two nonnegative regularization parameters to balance these terms. $\boldsymbol{1} \in \mathbb{R}^m$ is a column vector with all $1^{'}$s.

The problem in (11) could be solved by Inexact Augmented Lagrangian Method (IALM). It is an iterative algorithm, and thus needs to first introduce two auxiliary variables $\boldsymbol{Z}_1$ and $\boldsymbol{Z}_2$ to make the objective function separable. The following problem is obtained:

$$\begin{aligned} & \min_{\boldsymbol{Z}, \boldsymbol{Z}_1, \boldsymbol{Z}_2} \mathrm{tr}(\boldsymbol{D}^{\mathrm{T}} \boldsymbol{Z}) + \lambda_1 \|\boldsymbol{Z}_1\|_* + \lambda_2 \|\boldsymbol{Z}_2\|_1 \\ & s.t. \; \boldsymbol{Z}_1 = \boldsymbol{Z}; \; \boldsymbol{Z}_2 = \boldsymbol{Z}; \; \boldsymbol{1}^{\mathrm{T}} \boldsymbol{Z} = \boldsymbol{1}^{\mathrm{T}}; \; \boldsymbol{Z}_2 \geq \boldsymbol{0}, \end{aligned} \tag{12}$$

where $\|\cdot\|_1$ is the ℓ_1-norm of a matrix to relax the $\|\cdot\|_0$. Then, the augmented Lagrangian function of (12) is as follows.

$$\begin{aligned} \mathcal{L}(\boldsymbol{Z}, \boldsymbol{Z}_1, \boldsymbol{Z}_2) = & \, tr(\boldsymbol{D}^{\mathrm{T}} \boldsymbol{Z}) + \lambda_1 \|\boldsymbol{Z}_1\|_* + \lambda_2 \|\boldsymbol{Z}_2\|_1 \\ & + \langle \boldsymbol{\Delta}_1, \boldsymbol{Z} - \boldsymbol{Z}_1 \rangle + \langle \boldsymbol{\Delta}_2, \boldsymbol{Z} - \boldsymbol{Z}_2 \rangle + \langle \boldsymbol{\delta}_3, \boldsymbol{1}^{\mathrm{T}} \boldsymbol{Z} - \boldsymbol{1}^{\mathrm{T}} \rangle \\ & + \frac{\eta}{2} (\|\boldsymbol{Z} - \boldsymbol{Z}_1\|_F^2 + \|\boldsymbol{Z} - \boldsymbol{Z}_2\|_F^2 + \|\boldsymbol{1}^{\mathrm{T}} \boldsymbol{Z} - \boldsymbol{1}^{\mathrm{T}}\|_2^2) \end{aligned} \tag{13}$$

where $\boldsymbol{\Delta}_1$, $\boldsymbol{\Delta}_2 \in \mathbb{R}^{m\times m}$ and $\boldsymbol{\delta}_3 \in \mathbb{R}^m$ are the Lagrange multipliers. η is a penalty parameter.

According to the IALM, the objective function converges with a sequence of closed form updating steps. The variable $\boldsymbol{Z}$, $\boldsymbol{Z}_1$, or $\boldsymbol{Z}_2$ is updated with other variables fixed. The detailed updating rules are presented as follows.

$$\begin{aligned} \boldsymbol{Z} = & \arg \min_{\boldsymbol{Z}} \mathcal{L}(\boldsymbol{Z}) = (2\eta \boldsymbol{I} + \eta \boldsymbol{1}\boldsymbol{1}^{\mathrm{T}})^{-1} (\eta(\boldsymbol{Z}_1 + \boldsymbol{Z}_2 \\ & + \boldsymbol{1}\boldsymbol{1}^{\mathrm{T}}) - \boldsymbol{\Delta}_1 - \boldsymbol{\Delta}_2 - \boldsymbol{D} - \boldsymbol{1}\boldsymbol{\delta}_3) \end{aligned} \tag{14}$$

Algorithm 1 The implementation of SDLA

```
Input: $\boldsymbol{X}, \lambda_1, \lambda_2, \rho, \alpha$.    Initial.: $k \leftarrow 0, \boldsymbol{D}^{(k)}$.
repeat
    Initial. $t \leftarrow 0$, $\boldsymbol{Z}^{(t)} = \boldsymbol{Z}_1^{(t)} = \boldsymbol{Z}_2^{(t)} = \boldsymbol{Z}^{(k)}$, $\eta^{(t)} = 1$,
    $\boldsymbol{\Delta}_1^{(t)} = \boldsymbol{\Delta}_2^{(t)} = \boldsymbol{0}$, $\boldsymbol{\delta}_3^{(t)} = \boldsymbol{0}$.
    while not converged do
        Update $\boldsymbol{Z}^{(t+1)}$ according to (14);
        Update $\boldsymbol{Z}_1^{(t+1)}$ according to (15);
        Update $\boldsymbol{Z}_2^{(t+1)}$ according to (16);
        $\boldsymbol{\Delta}_1^{(t+1)} \leftarrow \boldsymbol{\Delta}_1^{(t)} + \eta(\boldsymbol{Z}^{(t+1)} - \boldsymbol{Z}_1^{(t+1)})$;
        $\boldsymbol{\Delta}_2^{(t+1)} \leftarrow \boldsymbol{\Delta}_2^{(t)} + \eta(\boldsymbol{Z}^{(t+1)} - \boldsymbol{Z}_2^{(t+1)})$;
        $\boldsymbol{\delta}_3^{(t+1)} \leftarrow \boldsymbol{\delta}_3^{(t)} + \eta(\boldsymbol{1}^{\mathrm{T}}\boldsymbol{Z}^{(t+1)} - \boldsymbol{1}^{\mathrm{T}})$;
        $\eta^{(t+1)} \leftarrow \rho\eta^{(t)}$;
        $t \leftarrow t + 1$;
    end while
    $\boldsymbol{Z}^{(k)} \leftarrow \boldsymbol{Z}^{(t)}$;
    Update $\Theta$ in deep architecture (10);
    $\boldsymbol{D} \leftarrow d_{ij} = \Phi_{\mathrm{dSim}}(\Phi_{\mathrm{rep}}(\Theta; \boldsymbol{x}_i), \Phi_{\mathrm{rep}}(\Theta; \boldsymbol{x}_j))$;
    $\boldsymbol{D}^{(k+1)} = \boldsymbol{D}^{(k)} + \alpha\boldsymbol{D}$; // Update with memory
    $k \leftarrow k + 1$;
until Convergence criterion satisfied
Output: Optimal solution $\boldsymbol{Z}^* = \boldsymbol{Z}^{(k)}$.
```

where $\boldsymbol{Z}$ in (14) is computed by equating the partial derivative of (13) with respect to $\boldsymbol{Z}$ to zero.

$$\boldsymbol{Z}_1 = \arg\min_{\boldsymbol{Z}_1} \mathcal{L}(\boldsymbol{Z}_1) = \Gamma_{\lambda_1\eta^{-1}}(\boldsymbol{Z} + \frac{1}{\eta}\boldsymbol{\Delta}_1) \qquad (15)$$

$$\boldsymbol{Z}_2 = \arg\min_{\boldsymbol{Z}_2} \mathcal{L}(\boldsymbol{Z}_2) = \max(\mathscr{S}_{\lambda_2\eta^{-1}}(\boldsymbol{Z} + \frac{1}{\eta}\boldsymbol{\Delta}_2), \boldsymbol{0}) \qquad (16)$$

where Γ and $\mathscr{S}$ are singular value soft-thresholding and shrinkage-thresholding operator, respectively. In detail, for any matrix $\boldsymbol{A}$ and parameter γ, the form of analytic solution for $\mathscr{S}$ is as follows.

$$\mathscr{S}_\gamma(\boldsymbol{A}) = \mathrm{sign}(\boldsymbol{A}) \max(|\boldsymbol{A}| - \gamma, 0) \qquad (17)$$

Then, we have the definition of Γ as

$$\Gamma_\gamma(\boldsymbol{A}) = \boldsymbol{U}\mathscr{S}_\gamma(\boldsymbol{\Lambda})\boldsymbol{V}^{\mathrm{T}} \qquad (18)$$

where $\boldsymbol{A} = \boldsymbol{U}\boldsymbol{\Lambda}\boldsymbol{V}^{\mathrm{T}}$ is the singular value decomposition.

4 Implementation Framework

In summary, Alg. 1 shows the steps of detailed implementation of the SDLA model in (1). The algorithm should not be terminated until the change of objective value is smaller than a pre-defined threshold (10^{-1} in our experiments). In addition, we initialize $\boldsymbol{D}$ by Euclidean distance.

5 Experimental Results and Analysis

In this section, we evaluate the performance of SDLA for selecting prototypes on several illustrative problems.

5.1 Clustering via Prototypes

To examine the performance of our proposed framework, we consider the problem of text clustering using prototypes that act as cluster centers. We compare our SDLA mainly with prototype selection based clustering methods, including AP [Frey and Dueck, 2007], Kmeans [Duda *et al.*, 2012], Kmedoids [Kaufman and Rousseeuw, 1987], Spectral Clustering (SC) [Ng *et al.*, 2002], LSC [Chen and Cai, 2011], N-SHLRR [Yin *et al.*, 2016] and DS3 [Elhamifar *et al.*, 2016]. To further verify the effectiveness of joint learning, we also compare SDLA with *Dis*-SDLA, which is our assignment model in (3) with given dissimilarity metric[1]. Note that many selection methods (e.g. SSDS) cannot be used for clustering.

The standard document collections **TDT2**[2] [Cai *et al.*, 2005] is used for this task, which consists of 9 groups of experiments corresponding to different cluster numbers. For each given cluster number, 30 tests are conducted on different randomly chosen clusters and the average performance is computed. The metric, clustering accuracy (AC), is used to measure the quality of prototypes. Table 1 shows the results of those methods, which prove that the proposed method works well with clustering task using prototypes. This is due to the fact that SDLA selects the most representative prototypes, thus improving grouping performance.

5.2 Evaluation by Classification

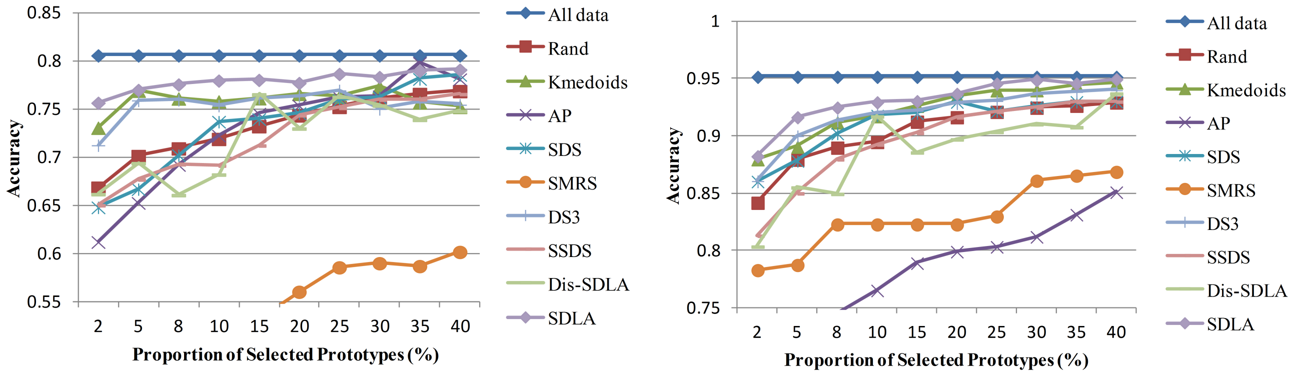
To further evaluate the discrimination of selected prototypes, we employ a classifier to compare the classification results when working on prototype set and source set. Here, we choose the 1-Nearest Neighbor (1-NN) classifier since it is parameter free and the results will be easily reproducible [Fan *et al.*, 2017]. Because the optimal number of prototypes is unknown, we compare each prototype selection method, including no selection (All data), random selection of samples (Rand), Kmedoids [Kaufman and Rousseeuw, 1987], AP [Frey and Dueck, 2007], SDS [Cong *et al.*, 2011], SMRS [Elhamifar *et al.*, 2012], DS3 [Elhamifar *et al.*, 2016], SSDS [Wang *et al.*, 2017], *Dis*-SDLA and SDLA, with varying numbers of selected prototypes. To this end, we consider the problems of scene categorization and handwriting recognition. The experiments are conducted on the **Fifteen Scene Categories dataset** [Lazebnik *et al.*, 2006] and the **USPS digit dataset** [Hull, 1994], which consist of 15 and 10 classes, respectively. We randomly select 80% of images in each class to form the source training set and use the rest for testing. For scene images, we take the 4096-dimensional CNN features [Simonyan and Zisserman, 2014] as input. Then a subset of training set is selected for 1-NN classification, whose averaged results over 6 times of execution with different training set selections are shown in Figure 5. Obviously, SDLA is the most closest to the ideal performance of using all data. This comes from the fact that SDLA effectively removes confusing samples by selecting the most discriminative prototypes.

[1] We choose the same dissimilarity metric with DS3.

[2] http://www.cad.zju.edu.cn/home/dengcai/Data/TextData.html

# cluster	AP	Kmeans	Kmedoids	SC	LSC	NSHLRR	DS3	*Dis*-SDLA	SDLA
2	77.18±1.10	57.74±1.39	96.72±0.90	79.34±2.12	65.31±2.09	86.65±1.28	96.72±1.55	95.39±1.95	**97.06±0.83**
3	68.28±1.91	62.74±2.80	50.96±2.14	78.11±1.42	94.36±1.48	89.35±2.33	97.45±0.96	96.23±1.15	**97.72±1.01**
4	63.95±2.79	64.47±1.95	65.79±1.65	86.65±1.03	71.84±2.84	82.62±1.93	**99.47±0.67**	97.91±0.75	98.12±0.74
5	60.41±1.39	45.11±3.65	60.27±2.17	68.59±3.70	71.95±1.03	**78.35±1.27**	71.80±2.04	72.82±3.09	76.55±2.82
6	63.52±1.69	66.05±2.31	69.48±3.95	74.73±1.03	**87.68±1.43**	74.09±2.38	74.93±1.76	73.21±1.79	80.36±1.18
7	65.89±1.48	56.65±1.40	53.34±3.63	62.05±2.70	65.78±3.75	70.77±2.27	70.65±1.66	70.23±1.65	**76.22±1.16**
8	52.37±3.11	68.49±1.49	58.06±3.95	71.66±2.34	70.02±2.58	71.69±1.22	70.92±3.75	71.29±2.24	**78.18±1.51**
9	59.78±3.70	68.51±2.89	65.95±2.96	69.30±2.55	69.66±2.14	70.13±1.14	69.45±3.26	69.10±3.84	**75.50±2.25**
10	60.11±1.81	52.83±3.24	58.41±2.92	76.81±1.35	72.30±1.19	79.17±1.25	82.51±1.61	81.33±1.47	**86.68±1.35**
ave.	63.50±2.11	60.30±2.35	64.33±2.69	74.10±2.02	74.32±2.06	78.09±1.67	81.43±1.91	80.77±1.99	**85.15±1.42**

Table 1: Clustering accuracy (AC) (%) of different methods on TDT2 Corpus.



(a)

(b)

Figure 5: 1-NN classification results of different prototype selection methods on different datasets. (a) Fifteen Scene Categories dataset. (b) USPS digit dataset.

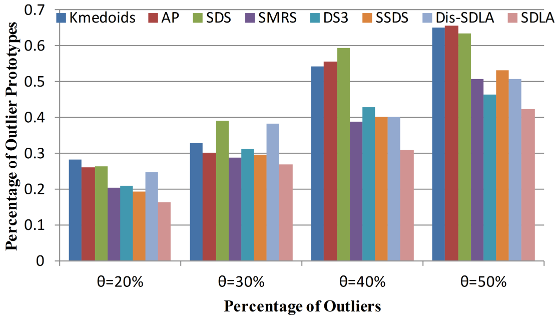


Figure 6: Percentage of outliers among the prototypes selected by different methods as a function of the fraction of outliers.

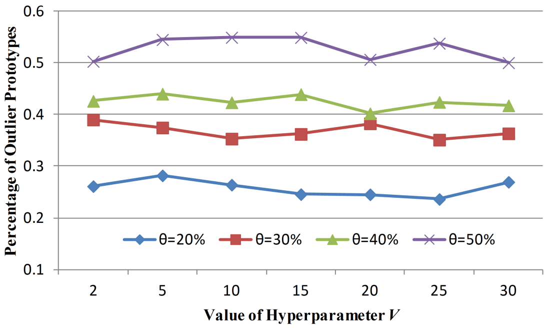


Figure 7: Percentage of outliers among the prototypes selected by *Dis*-SDLA as a function of hyperparameter υ.

5.3 Robustness to Outliers

To evaluate the performance of SDLA for rejecting outliers, we form a dataset of 9000 images, of which $1-\theta$ fraction are randomly selected from the **Extended YaleB face database**, and the remaining, corresponding to outliers, are random images downloaded from the internet. For $\theta \in \{20\%, 30\%, 40\%, 50\%\}$, we run SDLA as well as Kmedoids, AP, SDS, SMRS, DS3, SSDS and *Dis*-SDLA to select roughly 300 prototypes from the dataset. Figure 6 shows the percentage of outliers among the selected prototypes as we increase the number of outliers in the dataset. Obviously, SDLA results in less outliers compared with other methods. To facilitate the parameter tuning, we run Algorithm 1 with $\lambda_i = \lambda_0/\upsilon$, where $i = 1, 2$, λ_0 is computed from $\boldsymbol{X}$ [Elhamifar *et al.*, 2012], and $\upsilon \in [2, 30]$. Actually, the sensibility of SDLA degenerates into that of *Dis*-SDLA. Figure 7 presents the parameter analysis results for several values of υ.

6 Conclusion

In this work, we introduced a SDLA framework to select representative and discriminative prototypes. Promising experimental results show the effectiveness of SDLA.

Acknowledgments

This work was supported in part by the National Key R&D Program of China under Grant 2016YFB0800404 and the National Natural Science Foundation of China under Grants 61532005, 61332012, and 61572068, and in part by the Fundamental Research Funds for the Central Universities under Grant 2017YJS056.

References

[Bengio *et al.*, 2013] Yoshua Bengio, Aaron Courville, and Pascal Vincent. Representation learning: A review and new perspectives. *TPAMI*, 35(8):1798–1828, 2013.

[Cai *et al.*, 2005] Deng Cai, Xiaofei He, and Jiawei Han. Document clustering using locality preserving indexing. *TPAMI*, 17(12):1624–1637, 2005.

[Chan, 1987] Tony F Chan. Rank revealing QR factorizations. *Linear algebra and its applications*, 88:67–82, 1987.

[Chen and Cai, 2011] Xinlei Chen and Deng Cai. Large scale spectral clustering with landmark-based representation. In *AAAI*, pages 313–318, 2011.

[Cong *et al.*, 2011] Yang Cong, Junsong Yuan, and Ji Liu. Sparse reconstruction cost for abnormal event detection. In *CVPR*, pages 3449–3456, 2011.

[Cong *et al.*, 2017] Yang Cong, Ji Liu, Gan Sun, Quanzeng You, Yuncheng Li, and Jiebo Luo. Adaptive greedy dictionary selection for web media summarization. *TIP*, 26(1):185–195, 2017.

[Dornaika and Aldine, 2015] Fadi Dornaika and I Kamal Aldine. Decremental sparse modeling representative selection for prototype selection. *Pattern Recognit.*, 48(11):3714–3727, 2015.

[Duda *et al.*, 2012] Richard O Duda, Peter E Hart, and David G Stork. *Pattern classification*. John Wiley & Sons, 2012.

[Elhamifar and Kaluza, 2017] Ehsan Elhamifar and MCDP Kaluza. Online summarization via submodular and convex optimization. In *CVPR*, pages 1783–1791, 2017.

[Elhamifar *et al.*, 2012] Ehsan Elhamifar, Guillermo Sapiro, and Rene Vidal. See all by looking at a few: Sparse modeling for finding representative objects. In *CVPR*, pages 1600–1607, 2012.

[Elhamifar *et al.*, 2016] Ehsan Elhamifar, Guillermo Sapiro, and S Shankar Sastry. Dissimilarity-based sparse subset selection. *TPAMI*, 38(11):2182–2197, 2016.

[Fan *et al.*, 2017] Mingyu Fan, Xiaojun Chang, and Dacheng Tao. Structure regularized unsupervised discriminant feature analysis. In *AAAI*, pages 1870–1876, 2017.

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

[Garcia *et al.*, 2012] Salvador Garcia, Joaquin Derrac, Jose Cano, and Francisco Herrera. Prototype selection for nearest neighbor classification: Taxonomy and empirical study. *TPAMI*, 34(3):417–435, 2012.

[Hull, 1994] Jonathan J. Hull. A database for handwritten text recognition research. *TPAMI*, 16(5):550–554, 1994.

[Kaufman and Rousseeuw, 1987] Leonard Kaufman and Peter Rousseeuw. Clustering by means of medoids. In *Y. Dodge (Ed.) Statistical Data Anal. Based on the ℓ_1-norm and Related Methods*, pages 405–416, 1987.

[Krause *et al.*, 2008] Andreas Krause, H Brendan McMahan, Carlos Guestrin, and Anupam Gupta. Robust submodular observation selection. *J. Mach. Learn. Research*, 9:2761–2801, 2008.

[Lazebnik *et al.*, 2006] S. Lazebnik, C. Schmid, and J. Ponce. Beyond bags of features: Spatial pyramid matching for recognizing natural scene categories. In *CVPR*, pages 2169–2178, 2006.

[Lin *et al.*, 2018] Liang Lin, Keze Wang, Deyu Meng, Wangmeng Zuo, and Lei Zhang. Active self-paced learning for cost-effective and progressive face identification. *TPAMI*, 40(1):7–19, 2018.

[Meng *et al.*, 2016] Jingjing Meng, Hongxing Wang, Junsong Yuan, and Yap-Peng Tan. From keyframes to key objects: Video summarization by representative object proposal selection. In *CVPR*, pages 1039–1048, 2016.

[Nellore and Ward, 2015] Abhinav Nellore and Rachel Ward. Recovery guarantees for exemplar-based clustering. *Inform. and Computat.*, 245:165–180, 2015.

[Ng *et al.*, 2002] Andrew Y Ng, Michael I Jordan, and Yair Weiss. On spectral clustering: Analysis and an algorithm. In *NIPS*, pages 849–856, 2002.

[Okura *et al.*, 2017] Shumpei Okura, Yukihiro Tagami, Shingo Ono, and Akira Tajima. Embedding-based news recommendation for millions of users. In *KDD*, pages 1933–1942, 2017.

[Simonyan and Zisserman, 2014] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv:1409.1556*, 2014.

[Song *et al.*, 2017] Hyun Oh Song, Stefanie Jegelka, Vivek Rathod, and Kevin Murphy. Deep metric learning via facility location. In *CVPR*, pages 5382–5390, 2017.

[Vincent *et al.*, 2008] Pascal Vincent, Hugo Larochelle, Yoshua Bengio, and Pierre-Antoine Manzagol. Extracting and composing robust features with denoising autoencoders. In *ICML*, pages 1096–1103, 2008.

[Wang and Zhang, 2013] Shusen Wang and Zhihua Zhang. Improving CUR matrix decomposition and the nyström approximation via adaptive sampling. *J. Mach. Learn. Res.*, 14(9):2729–2769, 2013.

[Wang *et al.*, 2017] Hongxing Wang, Yoshinobu Kawahara, Chaoqun Weng, and Junsong Yuan. Representative selection with structured sparsity. *Pattern Recognit.*, 63:268–278, 2017.

[Yin *et al.*, 2016] Ming Yin, Junbin Gao, and Zhouchen Lin. Laplacian regularized low-rank representation and its applications. *TPAMI*, 38(3):504–517, 2016.

[Zhang *et al.*, 2018] Xingxing Zhang, Zhenfeng Zhu, Yao Zhao, and Dongxia Chang. Learning a general assignment model for video analytics. *TCSVT*, DOI:10.1109/TCSVT.2017.2713480, in press, 2018.

[Zhuang *et al.*, 2012] Liansheng Zhuang, Haoyuan Gao, Zhouchen Lin, Yi Ma, Xin Zhang, and Nenghai Yu. Nonnegative low rank and sparse graph for semi-supervised learning. In *CVPR*, pages 2328–2335, 2012.