SELF-PACED PROBABILISTIC PRINCIPAL COMPONENT ANALYSIS FOR DATA WITH OUTLIERS

Bowen Zhao[†,‡], *Xi Xiao*[†,‡], *Wanpeng Zhang*[†,‡], *Bin Zhang*[‡], *Guojun Gan*[*], *Shutao Xia*[†,‡]

[†]Tsinghua Shenzhen International Graduate School, Tsinghua University, Shenzhen, China
[‡]Peng Cheng Laboratory, Shenzhen, China
[*]Department of Mathematics, University of Connecticut, USA

ABSTRACT

Principal Component Analysis (PCA) is a popular tool for dimension reduction and feature extraction in data analysis. Probabilistic PCA (PPCA) extends the standard PCA by using a probabilistic model. However, both standard PCA and PPCA are not robust, as they are sensitive to outliers. To alleviate this problem, we propose a novel method called Self-Paced Probabilistic Principal Component Analysis (SP-PPCA) by introducing the Self-Paced Learning mechanism into PPCA. Furthermore, we design the corresponding optimization algorithm based on an alternative search strategy and an expectation-maximization algorithm, so that SP-PPCA uses an iterative procedure to find the optimal projection vectors and filter out outliers. Experiments on both synthetic data and real data demonstrate that SP-PPCA is more robust than the baselines.

***Index Terms*—** Probabilistic Principal Component Analysis, Self-Paced Learning, Robustness, Outliers

1. INTRODUCTION

Principal Component Analysis (PCA) [1] is a key method for dimension reduction and feature extraction [2, 3], which play an important role in machine learning [4], computer vison [5], genetics [6] and so on. High dimensional data poses great challenges in effective data analysis and brings a huge computing burden. PCA helps to mitigate these problems by attempting to represent high-dimensional data with principal components.

PCA can be defined in two ways. However, both ways are algebraic and lack probabilistic explanation for the observed data [7]. To overcome this shortcoming, Tipping and Bishop proposed a probabilistic model for PCA, called Probabilistic PCA (PPCA) [7]. PPCA has several advantages over standard PCA. For example, PPCA can deal with missing values in the observed data due to the underlying Expectation-Maximization (EM) algorithm. However, both PCA and PPCA inherently suffer from a drawback: they are not robust, that is, they are severely affected by outliers [8–10]. In the presence of outliers, the principal components obtained by standard PCA and PPCA are greatly deviated from the real direction. Consequently, we cannot get the main information of the data exactly. Subsequent analysis based on these principal components may lead to unsatisfied results.

By imitating human or animal learning, Self-Paced Learning (SPL) [11] first starts with simple samples of learning tasks, then incorporates complex examples into the training process step by step [12]. SPL has been successfully applied to object detection [13, 14], matrix factorization [15], mixture of regression [16]. In particular, it has been shown that SPL has the ability to reduce the effect of outliers [17, 18].

In order to improve the robustness of PPCA, we incorporate Self-Paced Learning mechanism into it, and propose a novel model called Self-Paced Probabilistic Principal Component Analysis (SP-PPCA). Based on PPCA, we design a new objective function by introducing additional parameters about self-paced learning. We design a corresponding optimization algorithm to learn the original parameters of PPCA and the additional parameters based on an alternative search strategy and an expectation-maximization algorithm. The proposed method attempts to learn from the clean training data gradually, and simultaneously prevents outliers from affecting the training process.

In summary, our main contributions in this paper include:

- To effectively eliminate the impact of outliers, we introduce the Self-Paced Learning mechanism into Probabilistic PCA. To the best of our knowledge, this is the first time of using SPL for PPCA.
- We propose a novel method named SP-PPCA from a probabilistic perspective and derive a corresponding optimization algorithm based on an expectation-maximization algorithm and an alternative search strategy.
- The experimental results based on both synthetic and real data show that the proposed method can obtain more accurate projection vectors from the contaminated data.

The remainder of this paper is organized as follows. In section 2, we briefly describe the Probabilistic PCA. In section 3, we introduce the proposed method Self-Paced Probabilistic PCA in detail. In section 4, we show some numerical experiments on synthetic data and real data to demonstrate the performance of the proposed method. Finally, we summarize the paper in section 5.

2. PROBABILISTIC PCA

PCA can be defined as the orthogonal projection of the data onto a lower dimensional linear space called the principal subspace, where the variance of the projected data is maximized [2,19]. An equivalent definition is the linear projection that minimizes the mean squared distance between the data points and their projections [1]. The above two definitions are algebraic and lack probabilistic explanation for the observed data. PCA can also be expressed as the maximum likelihood solution of a probabilistic latent variable model [2]. The probabilistic version of PCA is known as Probabilistic PCA (PPCA) [7].

In Probabilistic PCA, an M-dimensional vector of latent variable $\mathbf{z}$ corresponding to the principal component subspace is used. The prior distribution of $\mathbf{z}$ is assumed to be: $p(\mathbf{z}) = \mathcal{N}(\mathbf{z}|\mathbf{0}_M, \mathbf{I}_M)$. The D-dimensional observed data vector $\mathbf{x}$ is formulated by a linear combination of the latent variable $\mathbf{z}$ plus noise $\boldsymbol{\epsilon}$:

$$\mathbf{x} = \mathbf{W}\mathbf{z} + \boldsymbol{\mu} + \boldsymbol{\epsilon}, \tag{1}$$

where $\mathbf{W}$ is a $D \times M$ matrix that relates the observation vector $\mathbf{x}$ to the latent variable $\mathbf{z}$; the D-dimensional vector $\boldsymbol{\mu}$ allows the model to have non-zero mean; $\boldsymbol{\epsilon}$ is a D-dimensional Gaussian-distributed variable, i.e., $\boldsymbol{\epsilon} \sim \mathcal{N}(\mathbf{0}_D, \sigma^2 \mathbf{I}_D)$. Hence, equation (1) induces the conditional distribution of the observed data vector $\mathbf{x}$: $p(\mathbf{x}|\mathbf{z}) = \mathcal{N}(\mathbf{x}|\mathbf{W}\mathbf{z} + \boldsymbol{\mu}, \sigma^2 \mathbf{I}_D)$. By integrating out the latent variables, the marginal distribution of the observed variable is obtained: $p(\mathbf{x}) = \int p(\mathbf{x}|\mathbf{z})p(\mathbf{z})\mathrm{d}\mathbf{z} = \mathcal{N}(\mathbf{x}|\boldsymbol{\mu}, \mathbf{C})$, where $\mathbf{C}$ is a $D \times D$ covariance matrix defined by $\mathbf{C} = \mathbf{W}\mathbf{W}^{\mathrm{T}} + \sigma^2 \mathbf{I}_D$. To improve the efficiency of latter calculations, we notice the fact that $\mathbf{C}^{-1} = \sigma^{-2}\mathbf{I}_D - \sigma^{-2}\mathbf{W}\mathbf{M}^{-1}\mathbf{W}^{\mathrm{T}}$, where $\mathbf{M}$ is an $M \times M$ matrix defined by

$$\mathbf{M} = \mathbf{W}^{\mathrm{T}}\mathbf{W} + \sigma^2 \mathbf{I}_M. \tag{2}$$

In this way, the cost of evaluating $\mathbf{C}^{-1}$ is reduced from $O(D^3)$ to $O(M^3)$ [2]. Besides, the posterior distribution $p(\mathbf{z}|\mathbf{x})$ can be calculated using Bayes' rule as follows:

$$p(\mathbf{z}|\mathbf{x}) = \mathcal{N}(\mathbf{z}|\mathbf{M}^{-1}\mathbf{W}^{\mathrm{T}}(\mathbf{x} - \boldsymbol{\mu}), \sigma^2 \mathbf{M}^{-1}). \tag{3}$$

Finally, based on a dataset $\mathbf{X} = \{\mathbf{x}_n, n = 1, 2, \cdots, N\}$, where $\mathbf{x}_n$ is the vector presentation of observed data, the log-likelihood function is given by

$$\begin{aligned} \ln p(\mathbf{X}|\boldsymbol{\mu}, \mathbf{W}, \sigma^2) &= \sum_{n=1}^{N} \ln p\left(\mathbf{x}_n|\boldsymbol{\mu}, \mathbf{W}, \sigma^2\right) = -\frac{N}{2}\ln|\mathbf{C}| \\ &- \frac{ND}{2}\ln(2\pi) - \frac{1}{2}\sum_{n=1}^{N}\left(\mathbf{x}_n - \boldsymbol{\mu}\right)^{\mathrm{T}}\mathbf{C}^{-1}\left(\mathbf{x}_n - \boldsymbol{\mu}\right). \end{aligned} \tag{4}$$

The model parameters $\{\boldsymbol{\mu}, \mathbf{W}, \sigma^2\}$ can be estimated using the method of maximum likelihood. They have closed-form solutions, and can also be found through an EM algorithm [2, 7].

3. SELF-PACED PROBABILISTIC PCA

In this section, we present the proposed method and the corresponding optimization algorithm in detail.

3.1. Object Function

We incorporate the Self-Paced Learning mechanism into PPCA, and propose the novel method: Self-Paced Probabilistic Principal Component Analysis (SP-PPCA). The objective function of SP-PPCA is defined based on the above optimization problem (4) by introducing binary variables $v_n \in \{0, 1\}$ $(n = 1, 2, \cdots, N)$ and combining a sparse regularizer of v_n. Mathematically, the objective function is defined as:

$$\mathcal{L}(\mathbf{v}, \boldsymbol{\mu}, \mathbf{W}, \sigma^2) = -\sum_{n=1}^{N} v_n \ln p(\mathbf{x}_n|\boldsymbol{\mu}, \mathbf{W}, \sigma^2) - \beta\sum_{n=1}^{N} v_n, \tag{5}$$

where $\mathbf{v} = (v_1, v_2, \cdots, v_N)^{\mathrm{T}}$ and β is a hyper-parameter. The binary variable v_n indicates whether the n^{th} sample $\mathbf{x}_n$ is an outlier (if $v_n = 0$) or a clean sample (if $v_n = 1$). Note that, like the original model parameters $\{\boldsymbol{\mu}, \mathbf{W}, \sigma^2\}$, $\mathbf{v}$ also needs to be estimated from data. In SP-PPCA, the goal is to minimize equation (5). In subsection 3.2, we describe how to learn the parameters $\{\boldsymbol{\mu}, \mathbf{W}, \sigma^2\}$ and $\mathbf{v}$.

3.2. Optimization

We use an alternative search strategy to obtain approximate solution of the aforementioned optimization problem. Specifically, for a fixed $\mathbf{v}$, we update parameters $\{\boldsymbol{\mu}, \mathbf{W}, \sigma^2\}$; and given parameters $\{\boldsymbol{\mu}, \mathbf{W}, \sigma^2\}$, we estimate $\mathbf{v}$. A brief description of the algorithm is presented in Algorithm 1. In what follows, we describe each component in detail.

3.2.1. Optimization of $\{\boldsymbol{\mu}, \mathbf{W}, \sigma^2\}$

Given $\mathbf{v}$, we estimate the parameters $\{\boldsymbol{\mu}, \mathbf{W}, \sigma^2\}$ by solving the problem: $\min\limits_{\boldsymbol{\mu}, \mathbf{W}, \sigma^2} \mathcal{L}(\boldsymbol{\mu}, \mathbf{W}, \sigma^2; \mathbf{v})$, which is equivalent to the following optimization problem: $\max\limits_{\boldsymbol{\mu}, \mathbf{W}, \sigma^2} \sum_{n=1}^{N} v_n \ln p(\mathbf{x}_n|\boldsymbol{\mu}, \mathbf{W}, \sigma^2)$.

By setting the derivative of the above equation with respect to $\boldsymbol{\mu}$ to zero, we get:

$$\boldsymbol{\mu}_{\text{new}} = \frac{\sum_{n=1}^{N} v_n \mathbf{x}_n}{\sum_{n=1}^{N} v_n}, \tag{6}$$

and then substitute $\boldsymbol{\mu}$ with $\boldsymbol{\mu}_{\text{new}}$. Next, we use an EM algorithm to maximize the function with respect to $\mathbf{W}$ and σ^2. The complete-data log-likelihood function is given by:

$$\ln p(\mathbf{X}, \mathbf{Z}|\mathbf{v}, \boldsymbol{\mu}, \mathbf{W}, \sigma^2) = \sum_{n=1}^{N} v_n \Big\{ \ln p(\mathbf{x}_n|\mathbf{z}_n) + \ln p(\mathbf{z}_n) \Big\},$$

where $\mathbf{z}_n^{\mathrm{T}}$ is the n^{th} row of the matrix $\mathbf{Z}$. Then we calculate the expectation with respect to the posterior distribution of the latent variables as follows:

$$\begin{aligned} &\mathbb{E}[\ln p(\mathbf{X}, \mathbf{Z}|\mathbf{v}, \boldsymbol{\mu}, \mathbf{W}, \sigma^2)] = -\sum_{n=1}^{N} v_n \Bigg\{ \frac{D}{2}\ln\left(2\pi\sigma^2\right) + \\ &\frac{M}{2}\ln(2\pi) + \frac{1}{2}\operatorname{Tr}\left(\mathbb{E}\left[\mathbf{z}_n\mathbf{z}_n^{\mathrm{T}}\right]\right) + \frac{1}{2\sigma^2}\|\mathbf{x}_n - \boldsymbol{\mu}\|^2 + \frac{1}{2\sigma^2} \\ &\operatorname{Tr}\left(\mathbb{E}\left[\mathbf{z}_n\mathbf{z}_n^{\mathrm{T}}\right]\mathbf{W}^{\mathrm{T}}\mathbf{W}\right) - \frac{1}{\sigma^2}\mathbb{E}\left[\mathbf{z}_n\right]^{\mathrm{T}}\mathbf{W}^{\mathrm{T}}\left(\mathbf{x}_n - \boldsymbol{\mu}\right) \Bigg\}. \end{aligned}$$

From the above derivation, in the **E-step** of the EM algorithm, we compute

$$\begin{aligned} \mathbb{E}\left[\mathbf{z}_n\right] &= \mathbf{M}^{-1}\mathbf{W}^{\mathrm{T}}\left(\mathbf{x}_n - \boldsymbol{\mu}_{\text{new}}\right), \\ \mathbb{E}\left[\mathbf{z}_n\mathbf{z}_n^{\mathrm{T}}\right] &= \sigma^2\mathbf{M}^{-1} + \mathbb{E}\left[\mathbf{z}_n\right]\mathbb{E}\left[\mathbf{z}_n\right]^{\mathrm{T}}, \end{aligned} \tag{7}$$

where $\mathbf{M}$ is defined by equation (2). The above equations can be obtained easily using the posterior distribution (3) of latent variables. Then, in the **M-step**, by setting the derivatives with respect to $\mathbf{W}$ and σ^2 to zero respectively, we obtain the M-step re-estimation solutions:

$$\mathbf{W}_{\text{new}} = \left[\sum_{n=1}^{N} v_n\left(\mathbf{x}_n - \boldsymbol{\mu}_{\text{new}}\right)\mathbb{E}\left[\mathbf{z}_n\right]^{\mathrm{T}}\right]\left[\sum_{n=1}^{N} v_n\mathbb{E}\left[\mathbf{z}_n\mathbf{z}_n^{\mathrm{T}}\right]\right]^{-1}, \tag{8}$$

$$\begin{aligned} \sigma^2_{\text{new}} &= \frac{1}{D\sum_{n=1}^{N} v_n}\sum_{n=1}^{N} v_n\Big\{ \|\mathbf{x}_n - \boldsymbol{\mu}_{\text{new}}\|^2 \\ &+ \operatorname{Tr}\left(\mathbb{E}\left[\mathbf{z}_n\mathbf{z}_n^{\mathrm{T}}\right]\mathbf{W}_{\text{new}}^{\mathrm{T}}\mathbf{W}_{\text{new}}\right) - 2\mathbb{E}\left[\mathbf{z}_n\right]^{\mathrm{T}}\mathbf{W}_{\text{new}}^{\mathrm{T}}\left(\mathbf{x}_n - \boldsymbol{\mu}_{\text{new}}\right) \Big\}. \end{aligned} \tag{9}$$

From equations (6), (8) and (9), only data points indicated as "clean" can affect the parameters $\{\boldsymbol{\mu}, \mathbf{W}, \sigma^2\}$. In other words, the projection vectors obtained by SP-PPCA are rarely influenced by outliers.

Algorithm 1 : Self-Paced Probabilistic PCA

```
Input: Dataset $\mathbf{X} = \{\mathbf{x_n}, n = 1, 2, \cdots, N\}$, hyper-parameter $\eta$.
Output: Model parameters $\boldsymbol{\mu}$, $\mathbf{W}$, $\sigma^2$.
1: Initialize $\boldsymbol{\mu}$, $\mathbf{W}$, $\sigma^2$.
2: Initialize $\beta$.
3: repeat
4:     Update $\boldsymbol{\mu}$, $\mathbf{W}$, $\sigma^2$ by the EM algorithm :
5:         Calculate $\boldsymbol{\mu}$ by equation (6).
6:         repeat
7:             E-step: calculate $\mathbb{E}[\mathbf{z}_n]$, $\mathbb{E}[\mathbf{z}_n\mathbf{z}_n^{\mathrm{T}}]$ by equation (7).
8:             M-step: calculate $\mathbf{W}$, $\sigma^2$ by equations (8) and (9).
9:         until convergence
10:    Update $\mathbf{v}$ by equation (10).
11:    Update $\beta$, $\beta := \eta\beta$.
12: until convergence
13: return $\boldsymbol{\mu}$, $\mathbf{W}$, $\sigma^2$.
```

3.2.2. *Optimization of* $\mathbf{v}$

Fixing parameters $\{\boldsymbol{\mu}, \mathbf{W}, \sigma^2\}$, we estimate $\mathbf{v}$ by solving the following problem: $\min\limits_{v_n \in \{0,1\}} \mathcal{L}(\mathbf{v}; \boldsymbol{\mu}, \mathbf{W}, \sigma^2)$, which is equivalent to: $\min\limits_{v_n \in \{0,1\}} \sum_{n=1}^{N} v_n(l_n - \beta)$, where $l_n = -\ln p(\mathbf{x}_n|\boldsymbol{\mu}, \mathbf{W}, \sigma^2)$. Thus the solution of $\mathbf{v}$ can be obtained by:

$$v_n = \begin{cases} 0, & l_n > \beta, \\ 1, & l_n \leq \beta. \end{cases} \tag{10}$$

Then, the following two important questions can be answered now. (a) Why can $\mathbf{v}$ be viewed as the outlier indicators? (b) What role does the hyper-parameter β play?

On the basis of equation (10), we see that $\mathbf{v}$ can be estimated simply by using β as a threshold. When l_n is larger than β, v_n is set to 0. Since outliers are usually located far from the data center, they generally have lower likelihood according to equation (4). Thus, the training sample $\mathbf{x}_n$ is more likely to be an outlier if it has a larger value of l_n, i.e., lower likelihood. Therefore, it is reasonable to use $\mathbf{v}$ to indicate outliers in the training procedure.

It can be seen that β is crucial. If β is small, the optimization problem prefers to only consider relatively "cleaner" samples with high likelihood; on the contrary, most of samples (maybe also include outliers) are introduced if β is very large. In our implementation, we use a heuristic strategy to update β, as shown in Algorithm 1. We increase the value of β step by step, specifically, it is updated through a factor η, $\beta := \eta\beta$. This strategy is expected to collect clean data for training gradually.

3.2.3. *Summary*

The overall algorithm is shown in Algorithm 1. We update the parameters $\mathbf{v}$ and $\{\boldsymbol{\mu}, \mathbf{W}, \sigma^2\}$ alternatively by using an iterative procedure. In the proposed method SP-PPCA, the two processes work together by looking for the optimal projection vectors and filtering out outliers.

Once we get the final results $\{\boldsymbol{\mu}_*, \mathbf{W}_*, {\sigma^2}_*\}$, a D-dimensional point $\mathbf{x}$ in the data space can be represented by the corresponding posterior mean and covariance in the latent space according to equation (3) [2]. The mean is obtained by: $\mathbb{E}[\mathbf{z}|\mathbf{x}] = \mathbf{M}_*^{-1}\mathbf{W}_*^{\mathrm{T}}(\mathbf{x} - \boldsymbol{\mu}_*)$, where $\mathbf{M}_* = \mathbf{W}_*^{\mathrm{T}}\mathbf{W}_* + \sigma_*^2\mathbf{I}_M$. We also can use $\hat{\mathbf{x}}$ obtained by $\hat{\mathbf{x}} = \mathbf{W}\mathbb{E}[\mathbf{z}|\mathbf{x}] + \boldsymbol{\mu}_*$ to reconstruct the original data point $\mathbf{x}$.

3.3. Convergence Discussion

In subsection 3.2.1, we fix $\mathbf{v}$ and update $\{\boldsymbol{\mu}, \mathbf{W}, \sigma^2\}$ by an EM algorithm. It has been confirmed that the EM algorithm changes the model parameters in a direction which reduces the negative log likelihood (unless it is already at a minimum) [2]. Therefore, we have

$$\mathcal{L}(\mathbf{v}^{t-1}, \boldsymbol{\mu}^t, \mathbf{W}^t, {\sigma^2}^t) \leq \mathcal{L}(\mathbf{v}^{t-1}, \boldsymbol{\mu}^{t-1}, \mathbf{W}^{t-1}, {\sigma^2}^{t-1}). \tag{11}$$

where t stands for the number of epoch.

In subsection 3.2.2, we fix $\{\boldsymbol{\mu}, \mathbf{W}, \sigma^2\}$ and update $\mathbf{v}$. The second derivatives of $\mathcal{L}(\mathbf{v}; \boldsymbol{\mu}, \mathbf{W}, \sigma^2)$ for v_n are zero, indicating that $\mathcal{L}(\mathbf{v}; \boldsymbol{\mu}, \mathbf{W}, \sigma^2)$ is convex with respect to $\mathbf{v}$. Thus, we have

$$\mathcal{L}(\mathbf{v}^t, \boldsymbol{\mu}^t, \mathbf{W}^t, {\sigma^2}^t) \leq \mathcal{L}(\mathbf{v}^{t-1}, \boldsymbol{\mu}^t, \mathbf{W}^t, {\sigma^2}^t), \tag{12}$$

By substituting equation (11) back into equation (12), we have, $\forall t \geq 1$,

$$\mathcal{L}(\mathbf{v}^t, \boldsymbol{\mu}^t, \mathbf{W}^t, {\sigma^2}^t) \leq \mathcal{L}(\mathbf{v}^{t-1}, \boldsymbol{\mu}^{t-1}, \mathbf{W}^{t-1}, {\sigma^2}^{t-1}), \tag{13}$$

which indicates that the objective decreases in every iteration. Thus, it is guaranteed that Algorithm 1 converges to a stationary solution.

4. EXPERIMENTS

In this section, we first introduce the evaluation methodology, the baselines and the implementation details in the experiments. Then, we test the proposed method on two-dimensional data so that we could visually observe the projection vectors. Finally, we compare the performance of SP-PPCA with other methods on synthetic data and real data.

4.1. Settings

The goal of our work is to get the correct principal components that are not influenced much by outliers. We follow the evaluation method used in [20, 21]. Formally, we have a contaminated train dataset $\mathbf{X}_{\text{train}}$ and a clean test dataset $\mathbf{X}_{\text{test}}$. We perform dimension reduction on the contaminated data $\mathbf{X}_{\text{train}}$, and obtain projection vectors. Then we calculate the reconstruction error on the test data by $Error = \frac{||\mathbf{X}_{\text{test}} - \widehat{\mathbf{X}}_{\text{test}}||_F}{||\mathbf{X}_{\text{test}}||_F}$, where $\widehat{\mathbf{X}}_{\text{test}}$ is the recovered data for $\mathbf{X}_{\text{test}}$ based on the obtained projection vectors, $||\cdot||_F$ is the Frobenius norm. A small reconstruction error means that the projection vectors obtained by dimension reduction methods contain more favorable information about true data and less negative information about outliers.

The proposed algorithm SP-PPCA is tested with classical PCA, PPCA [7], PCP [22, 23], ROBPCA [24] and L_1-norm PCA [25] as baseline methods. The PCP algorithm decomposes the observed data into a low rank matrix and a sparse matrix. We treat the sparse matrix as noise part, and perform standard PCA on the low rank matrix. ROBPCA combines projection pursuit ideas with robust covariance estimator. L_1-norm PCA resorts to the L_1-norm to counteract outliers instead of the L_2-norm.

In our experiments, these methods are implemented with the default parameters. For the proposed method SP-PPCA, we initialize $\boldsymbol{\mu}$, $\mathbf{W}$ and σ^2 by the mean of samples, a random matrix, and scalar 1, respectively. Then, we run small number of iterations of the original PPCA algorithm and initialize β with the top 40 percent of l_n, $n = 1, 2, \cdots, N$. We set η to 1.01 in all of our experiments. The results reported are averaged over 10 trials.

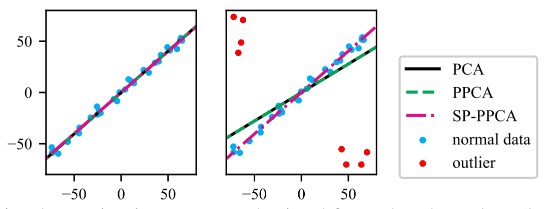

Fig. 1: The projection vectors obtained from the clean data (left) and the dirty data (right). The plotted data is centered.

Table 1: Average reconstruction errors for synthetic data (M is the number of principal components).

setting	N:300, D:500, M:10			N:300, D:200, M:5		
outlier rate	0.0	0.1	0.3	0.0	0.1	0.3
PCA	0.3075	0.6876	0.8938	0.4102	0.5162	0.7668
PPCA	0.3075	0.6888	0.8942	0.4102	0.5177	0.7700
PCP	0.3099	0.5161	0.8149	0.4109	0.4954	0.7458
ROBPCA	0.3094	0.3207	0.8938	0.4107	0.4113	0.7668
L_1PCA	0.3124	0.6454	0.8251	0.4118	0.4859	0.6575
SP-PPCA	0.3076	0.3086	0.3113	0.4108	0.4111	0.4130

4.2. Experiments on two-dimensional data

In the clean data set $\{x_n, y_n\}$, $\{x_n\}$ are sampled from a uniform distribution from 0 to 150, and $\{y_n\}$ are draw from $y_n = 0.8 \times x_n + 5 + \epsilon$, where the noise ϵ is generated from normal distribution $\mathcal{N}(0, 3)$. Then we add some outliers to the clean data. We reduce the clean data and the dirty data to one dimension by PCA, PPCA and SP-PPCA, respectively.

Figure 1 plots projection vectors obtained by these algorithms on both the clean data (left) and the dirty data (right). We can see that the performance of these methods are similar on the clean data. However when it comes to the dirty data, PCA and PPCA are seriously affected by outliers. The outliers cause the projection vectors of PCA and PPCA to deviate from the correct direction. The results of these experiments show that the proposed method is more robust than the standard PCA and PPCA.

4.3. Experiments on low-rank matrices

We build a low-rank matrix as $\mathbf{X}_{raw} = \mathbf{U}\mathbf{V}^{\mathrm{T}} + \mathbf{E}$, where $\mathbf{U}$ (or $\mathbf{V}$) are taken as N (or D) cases from a multivariate normal distribution $\mathcal{N}(\mathbf{0}, \mathbf{I})$; $\mathbf{E}$ is an $N \times D$ noise matrix, each row of which are generated from multivariate normal distribution $\mathcal{N}(\mathbf{0}_D, \mathbf{I}_D)$. The dataset is divided into training set (70%) and test set (30%). Then the contaminated training data set $\mathbf{X}_{\text{train}}$ are constructed by replacing some normal samples in the original train set with outliers[1]. We construct a set of data with different sizes and various percentage of outliers. Then we run SP-PPCA and other algorithms on the occluded train data, and calculate reconstruction errors on the test data.

Table 1 shows the average reconstruction errors under different settings. We see that the different methods perform similarly on the clean data. However, PCA, PPCA are greatly impacted by the outliers in the contaminated data. PCP, ROBPCA and L_1-norm PCA are able to reduce the effect of outliers to some extent. The comparison results show that the proposed method SP-PPCA achieves lower reconstruction errors and is superior to the other methods.

[1]We adopt the definition of outliers in [26]: "regular outliers" are the observations outside the range $[Q_1 - k(Q_3 - Q_1), Q_3 + k(Q_3 - Q_1)]$, where $k \in (1.5,\ 3)$, Q_1 and Q_3 are the lower and upper quartiles respectively.

Table 2: Average reconstruction errors for ISOLET.

setting	M:10			M:30		
outlier rate	0.0	0.1	0.4	0.0	0.1	0.4
PCA	0.4094	0.4906	0.5596	0.3151	0.4437	0.5222
PPCA	0.4095	0.4903	0.5591	0.3154	0.4438	0.5229
PCP	0.4144	0.4591	0.5545	0.3280	0.3768	0.4998
ROBPCA	0.4123	0.4112	0.5588	0.3207	0.3205	0.5218
SP-PPCA	0.4123	0.4102	0.4128	0.3193	0.3166	0.3226

Table 3: Run-time of SP-PPCA and other methods on synthetic data (the first row) and real data (the second row) with 10% outlier. The run time are recorded in second (s).

setting	PCA	PPCA	PCP	ROBPCA	L_1PCA	SP-PPCA
N:300, D:200, M:5	0.02	0.03	14.95	0.44	60.27	1.42
M=10	0.01	0.09	176.14	2.28	—	17.23

4.4. Experiments on real dataset

We conduct experiments on ISOLET from UCI repository [27]. ISOLET is a dataset for spoken letter recognition with 7797 instances, and 617 attributes per sample. We randomly select 600 samples as the training set, 400 samples as the test set and contaminate the clean train data by some outliers. Then dimension reduction algorithms[2] are used to project the data to a low dimensional space. Finally, the reconstruction errors are calculated on the test data.

Table 2 presents the average reconstruction errors with different number of principal components and outlier rates. The performances of most methods deteriorate rapidly as the number of outliers increases. This is similar to the results of our experiments in section 4.3. The proposed method tends to remove possible outliers from the dirty data, thus it can get good results on the contaminated data.

4.5. Run-time Analysis

The runtime for these experiments is shown in Table 3. We see that SP-PPCA is more efficient and accurate than other robust methods such as L_1PCA and PCP, although it takes more time than standard PCA and PPCA due to the iterative mechanism.

5. CONCLUSION AND FUTURE WORK

In this paper, with the aim to build a more robust dimension reduction algorithm, we propose a novel method called SP-PPCA. The proposed method incorporates Self-Paced Learning mechanism into Probabilistic PCA. We develop the corresponding algorithm for optimizing the model parameters. We compare the proposed method with standard PCA and several variants on both synthetic and real data. The experiments demonstrate the effectiveness of our method. SP-PPCA is based on Gaussian distribution, other robust distribution, such as heavy-tailed distributions [28], can be adopted in SP-PPCA to improve the robustness in future.

Acknowledgements This work is supported in part by the National Natural Science Foundation of Guangdong Province (2018A030313422), the National Natural Science Foundation of China (61972219, 61771273), the National Key Research and Development Program of China (2018YFB1800204, 2018YFB1800600), the Key Research and Development Program for Guangdong Province (2019B010136001), the Science and Technology Planning Project of Guangdong Province LZC0023 and LZC0024, the R&D Program of Shenzhen (JCYJ20180508152204044), and the research fund of PCL Future Regional Network Facilities for Large-scale Experiments and Applications (PCL2018KP001).

[2]The results of L_1PCA is not reported here, as it cannot provide results within a reasonable amount of time.

6. REFERENCES

[1] Karl Pearson, "LIII. on lines and planes of closest fit to systems of points in space," *The London, Edinburgh, and Dublin Philosophical Magazine and Journal of Science*, vol. 2, no. 11, pp. 559–572, 1901.

[2] Christopher M Bishop, *Pattern recognition and machine learning*, springer, 2006.

[3] Ian Jolliffe, *Principal component analysis*, Springer, 2011.

[4] Kevin P Murphy, *Machine learning: a probabilistic perspective*, MIT press, 2012.

[5] Fernando De la Torre and Michael J Black, "Robust principal component analysis for computer vision," in *Proceedings Eighth IEEE International Conference on Computer Vision. ICCV 2001*. IEEE, 2001, vol. 1, pp. 362–369.

[6] Alessandro Biffi, Christopher D Anderson, Michael A Nalls, Rosanna Rahman, Akshata Sonni, Lynelle Cortellini, Natalia S Rost, Mar Matarin, Dena G Hernandez, Anna Plourde, et al., "Principal-component analysis for assessment of population stratification in mitochondrial medical genetics," *The American Journal of Human Genetics*, vol. 86, no. 6, pp. 904–917, 2010.

[7] Michael E Tipping and Christopher M Bishop, "Probabilistic principal component analysis," *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, vol. 61, no. 3, pp. 611–622, 1999.

[8] I Stanimirova, M Daszykowski, and B Walczak, "Dealing with missing values and outliers in principal component analysis," *Talanta*, vol. 72, no. 1, pp. 172–178, 2007.

[9] Sven Serneels and Tim Verdonck, "Principal component analysis for data containing outliers and missing elements," *Computational Statistics & Data Analysis*, vol. 52, no. 3, pp. 1712–1727, 2008.

[10] Teng Zhang and Gilad Lerman, "A novel m-estimator for robust pca," *The Journal of Machine Learning Research*, vol. 15, no. 1, pp. 749–808, 2014.

[11] M Pawan Kumar, Benjamin Packer, and Daphne Koller, "Self-paced learning for latent variable models," in *Advances in Neural Information Processing Systems*, 2010, pp. 1189–1197.

[12] Deyu Meng, Qian Zhao, and Lu Jiang, "What objective does self-paced learning indeed optimize?," *arXiv preprint arXiv:1511.06049*, 2015.

[13] Dingwen Zhang, Deyu Meng, Long Zhao, and Junwei Han, "Bridging saliency detection to weakly supervised object detection based on self-paced curriculum learning," *arXiv preprint arXiv:1703.01290*, 2017.

[14] Enver Sangineto, Moin Nabi, Dubravko Culibrk, and Nicu Sebe, "Self paced deep learning for weakly supervised object detection," *IEEE transactions on pattern analysis and machine intelligence*, vol. 41, no. 3, pp. 712–725, 2019.

[15] Qian Zhao, Deyu Meng, Lu Jiang, Qi Xie, Zongben Xu, and Alexander G Hauptmann, "Self-paced learning for matrix factorization," in *Twenty-Ninth AAAI Conference on Artificial Intelligence*, 2015.

[16] Longfei Han, Dingwen Zhang, Dong Huang, Xiaojun Chang, Jun Ren, Senlin Luo, and Junwei Han, "Self-paced mixture of regressions.," in *IJCAI*, 2017, pp. 1816–1822.

[17] Yang Zhang, Qingtao Tang, Li Niu, Tao Dai, Xi Xiao, and Shu-Tao Xia, "Self-paced mixture of t distribution model," in *2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2018, pp. 2796–2800.

[18] Lu Jiang, "Web-scale multimedia search for internet video content," in *Proceedings of the 25th International Conference Companion on World Wide Web*. International World Wide Web Conferences Steering Committee, 2016, pp. 311–316.

[19] Harold Hotelling, "Analysis of a complex of statistical variables into principal components.," *Journal of educational psychology*, vol. 24, no. 6, pp. 417, 1933.

[20] Breton Minnehan and Andreas Savakis, "Grassmann manifold optimization for fast L_1-norm principal component analysis," *IEEE Signal Processing Letters*, vol. 26, no. 2, pp. 242–246, 2019.

[21] Fujiao Ju, Yanfeng Sun, Junbin Gao, Yongli Hu, and Baocai Yin, "Image outlier detection and feature extraction via L_1-norm-based 2d probabilistic pca," *IEEE Transactions on Image Processing*, vol. 24, no. 12, pp. 4834–4846, 2015.

[22] Wei Xiao, Xiaolin Huang, Jorge Silva, Saba Emrani, and Arin Chaudhuri, "Online robust principal component analysis with change point detection," *arXiv preprint arXiv:1702.05698*, 2017.

[23] Emmanuel J Candès, Xiaodong Li, Yi Ma, and John Wright, "Robust principal component analysis?," *Journal of the ACM (JACM)*, vol. 58, no. 3, pp. 11, 2011.

[24] Mia Hubert, Peter J Rousseeuw, and Karlien Vanden Branden, "Robpca: a new approach to robust principal component analysis," *Technometrics*, vol. 47, no. 1, pp. 64–79, 2005.

[25] Panos P Markopoulos, Sandipan Kundu, Shubham Chamadia, and Dimitris A Pados, "Efficient L_1-norm principal-component analysis via bit flipping," *IEEE Transactions on Signal Processing*, vol. 65, no. 16, pp. 4252–4264, 2017.

[26] J W Tuckey, *Exploratory Data Analysis*, 1977.

[27] Dheeru Dua and Casey Graff, "UCI machine learning repository," 2017.

[28] Jaakko Luttinen, Alexander Ilin, and Juha Karhunen, "Bayesian robust pca of incomplete data," *Neural processing letters*, vol. 36, no. 2, pp. 189–202, 2012.