Supervised Dimensionality Reduction Via Nonlinear Target Estimation

Josif Grabocka, Lucas Drumond, and Lars Schmidt-Thieme

Information Systems and Machine Learning Lab
Samelsonplatz 22, 31141 Hildesheim, Germany
{josif,ldrumond,schmidt-thieme}@ismll.uni-hildesheim.de

Abstract. Dimensionality reduction is a crucial ingredient of machine learning and data mining, boosting classification accuracy through the isolation of patterns via omission of noise. Nevertheless, recent studies have shown that dimensionality reduction can benefit from label information, via a joint estimation of predictors and target variables from a low-rank representation. In the light of such inspiration, we propose a novel dimensionality reduction which simultaneously reconstructs the predictors using matrix factorization and estimates the target variable via a dual-form maximum margin classifier from the latent space. The joint optimization function is learned through a coordinate descent algorithm via stochastic updates. Finally empirical results demonstrate the superiority of the proposed method compared to both classification in the original space (no reduction), or classification after unsupervised reduction.

Keywords: Machine Learning; Dimensionality Reduction; Feature Extraction; Matrix Factorization; Supervised Dimensionality Reduction

1 Introduction

Dimensionality reduction is an important ingredient of machine learning and data mining. The benefits of projecting data to latent spaces constitute in (i) converting large dimensionality datasets into feasible dimensions, but also (ii) improving the classification accuracy of small and medium datasets [1]. Via carefully tuned dimensionality reduction (aka feature extraction) we are able to retrieve the necessary patterns from the datasets, by leaving out the noise. Traditional dimensionality reduction, (as described in Section 2.1), has been focused on extracting features prior to classification. Such a mentality has been recently found to perform non-optimal [2, 3], since the features are not directly extracted/optimized for boosting classification. As a result there have been attempts to incorporate class supervision into feature extraction, (mentioned in Section 2.3), such that the latent features are guided to enforce the discernment/separation of instances belonging to opposite classes in the reduced space. Throughout this work we propose a principle, (details in Section 3.1), according to which dimensionality reduction should optimize the latent features through

the same optimization function as the final classification method, thereby ensuring that the classification accuracy in the latent space is optimized. Inspired by the accuracy success of SVMs, that is significantly inherent to the kernel trick approach, we propose a novel supervised dimensionality reduction that incorporates kernel-based classification in the reduced dimension (Section 3). The novelty relies on defining a joint dimensionality reduction via matrix factorization, in parallel to a dual-form kernel-based maximum margin classification in the latent space. The reduced data is simultaneously updated in a coordinate descent fashion in order to optimize both loss terms. Experimental results, (Section 5), demonstrate the superiority of the proposed method compared to both unsupervised dimensionality reduction and classification in the original space. The main contribution of this work are:

1. Define a supervised dimensionality reduction with a kernel-based target variable estimation, in addition to the matrix reconstruction loss term
2. Derive a coordinate descent algorithm which simultaneously learns the latent factors for both loss terms
3. Provide empirical results to demonstrate the superiority of the method

2 Related Work

2.1 Dimensionality Reduction

Dimensionality reduction is a field of computer science that focuses on extracting lower dimensionality features from datasets [1]. Numerous techniques exist for extracting features. Principal Component Analysis (PCA) is a famous approach involving orthogonal transformations and selecting the topmost principal components, which preserve necessary variance [4]. Alternatively, Singular Value Decomposition decomposes a dataset into latent unitary, nonnegative diagonal and conjugate transpose unitary matrices [1].

Further elaborations of dimensionality reductions involve nonlinear decomposition of data [5]. For instance kernel PCA replaces the linear operations of PCA through nonlinear mapping in a reproducing kernel Hilbert space [6]. The whole subfield of manifold learning elaborates, as well, on nonlinear projections. Specifically, Sammon's mappings preserves the structure of instance distances in the reduced space [7], while principal curves embed manifolds using standard geometric projections [8]. More nonlinear dimensionality algorithms are described in [9]. In addition, temporal dimensionality reduction have been proposed in scenarios where the time difference of observations is not evenly spaced [10].

2.2 Matrix Factorization

Matrix factorization refers to a family of decompositions which approximates a dataset as a product of latent matrices of typically lower dimensions. A generalization and categorization of the various proposed factorization models as applications of Bregman divergences was elaborated in [11]. The learning of the

decomposition is typically conducted by defining a l2-norm and updating the latent matrices via a stochastic gradient descent [12]. Matrix factorization has been applied in a range of domains, ranging from recommender systems where decomposition focuses on collaborative filtering of sparse user-item ratings [13], up to time series dimensionality reduction [14].

2.3 Supervised Dimensionality Reduction

In addition to the standard dimensionality reduction and matrix factorization, there has been attempts to utilize the labels information, therefore dictating a supervised projection. Fisher's linear discriminant analysis is a popular supervised projection method [15]. The classification accuracy loss objective functions occurring in literature vary from label least square regression [16], to generalized linear models [17], linear logistic regression [2], up to hinge loss [3]. Another study aimed at describing the target variable as being conditionally dependent on the features [18]. Other families of supervisions aim at preserving the neighborhood structure of intra-class instances [19], or links in a semi supervised scenarios [20]. In comparison to the aforementioned, we propose a supervised dimensionality reduction with a kernel-based classifier, by directly optimizing the dual formulation in the projected space.

3 Proposed Method

3.1 Principle

The method proposed in this study relies on the principle that feature extraction, analogously referred also as dimensionality reduction, should not be conducted "ad-hoc" or via particular heuristics. Most of the classification tasks have a unifying objective, which is to improve classification accuracy. In that context we are referring as "ad-hoc" to the family of feature extraction techniques that don't directly optimize their loss functions for classification accuracy. Stated else-wise, we believe that instance labels should guide the feature extraction, such that the utilization of the extracted features improves accuracy. In that perspective, we propose a feature extraction method which operates by optimizing a joint objective function composed of the feature extraction term and also the classification accuracy term. Further details will be covered in the coming Section 3. In comparison with similar feature extraction ideas reviewed in Section 2.3, which use linear classifiers in the optimization, we propose a novel method which learns a nonlinear SVMs over the projected space via jointly optimizing a dual form together with dimensionality reduction.

3.2 Matrix Factorization as Dimensionality Reduction

Matrix factorization is a dimensionality reduction technique which decomposes a dataset $X \in \mathbb{R}^{(n+n')\times m}$ matrix of n training instances and n' testing instances, per m features, into two smaller matrices of dimensions $U \in \mathbb{R}^{(n+n')\times d}$ and

$V \in \mathbb{R}^{d \times m}$ [12]. The latent/reduced projection of the original data X is the latent matrix U, where d is the dimensionality of the projected space. Typically d is much smaller than m, meaning that the dimensionality is reduced. Such decomposition is expressed in form of a reconstruction loss, denoted $F_R(X, U, V)$ and depicted in Equation 1. The optimization aims at computing latent matrices U, V such that their dot product approximates the original matrix X via an Euclidean distance (l_2 norm) loss. In addition to the l_2 reconstruction norm, we also add l_2 regularization terms weighted by factors λ_U, λ_V in order to avoid over-fitting.

$$\underset{U,V}{\operatorname{argmin}} \; F_R(X, U, V) = ||X - UV||^2 + \lambda_U ||U||^2 + \lambda_V ||V||^2 \tag{1}$$

Bias terms, $B_U \in \mathbb{R}^{(n+n') \times 1}, B_V \in \mathbb{R}^{1 \times m}$ are added to the reconstruction loss [12], such that each element of B_U incorporates the prior belief value of the respective instance, while each element of B_V the prior belief value of the respective feature. More concretely the loss can be expanded as a reconstruction of each cell $X_{i,j}$ as depicted by Equation 2.

$$\begin{aligned} \underset{U,V,B_U,B_V}{\operatorname{argmin}} \; F_R(X, U, V) = & \sum_{i=1}^{n+n'} \sum_{j=1}^{m} \left(X_{i,j} - \left(\sum_{k=1}^{d} U_{i,k} V_{k,j} + B_{U_i} + B_{V_j} \right) \right)^2 \\ & + \lambda_U \sum_{i=1}^{n+n'} \sum_{k=1}^{d} U_{i,k}^2 + \lambda_V \sum_{k=1}^{d} \sum_{j=1}^{m} V_{k,j}^2 \end{aligned} \tag{2}$$

3.3 Kernel-based Supervision of Dimensionality Reduction

Matrix factorization, as described in Section 3.2, is guided only by the reconstruction loss. Such approach doesn't take into consideration the classification accuracy impact of the extracted features, therefore the produced reduced dimensionality data is not optimized to improve accuracy. In order to overcome such a drawback, the so called supervised dimensionality reduction has been proposed by various authors [2]. The key commonalities of those supervised dimensionality methods rely on defining a joint optimization function, consisting of the reconstruction loss terms and the classification accuracy terms.

The typical classification accuracy loss term focuses on defining a classifier in the latent space, i.e. U, via a hyperplane defined by the weights vector W, such that the weights can correctly classify the training instances of U in order to match observed label Y. Equation 3 defines a cumulative joint optimization function using an introduced classification accuracy term, denoted $F_{CA}(Y, U, W)$. The trick of such a joint optimization constitutes on updating U simultaneously, in order to minimize both F_R and F_{CA} via gradient descent on both loss terms. The hyper parameter β is a switch which balances the impact of reconstruction vs classification accuracy.

$$F(X, Y, U, V, W) = \beta \, F_R(X, U, V) + (1 - \beta) \, F_{CA}(Y, U, W) \tag{3}$$

In comparison to previous approaches that propose linear models, in this study we propose a kernel-based **binary** classifier approach in the latent space U. Let us initially define the classification accuracy loss term, denoted $F_{CA}(Y, U, W)$, in Equation 4, in form of a maximum margin soft SVMs with hinge loss [21]. Such form of the SVMs is called the primal form. The parameter C scales the penalization of the instances violating the distances from the maximum margin. Please note that W_0 is the intercept bias term of the hyperplane weights vector W.

$$\begin{aligned} \underset{U,W}{\text{argmin}} \;\; & F_{CA}(Y,U,W) = \frac{1}{2}||W||^2 + C\sum_{i=1}^{n}\xi_i \\ \text{s.t:} \;\; & Y_i(\langle W, U_i\rangle + W_0) \geq 1 - \xi_i, \;\; i = 1, ..., n \\ & \xi_i \geq 0, \;\; i = 1, ..., n \end{aligned} \tag{4}$$

Unfortunately the primal form doesn't support kernels, therefore we have to convert the optimization functions into the dual form equation 5. In order to get rid of of the inequality constraint we apply Lagrange multipliers to include the inequalities by introducing dual variables α_i per instance and adding $\alpha_i \left(y_i(\langle W, U_i\rangle + W_0)\right)$ to the optimization function for all instance i. Then we solve the objective function for W and W_0 by equating the first derivative to zero. Putting the derived expressions of W and W_0 to the objective function, we obtain the so-called dual representation optimization:

$$\begin{aligned} \underset{U,\alpha}{\text{argmin}} \;\; & F_{CA}(Y,U,\alpha) = \frac{1}{2}\sum_{i=1}^{n}\sum_{l=1}^{n}\alpha_i\alpha_l Y_i Y_l \langle U_{i,*}, U_{l,*}\rangle - \sum_{i=1}^{n}\alpha_i \\ \text{s.t:} \;\; & 0 \leq \alpha_i \leq C, \;\; i = 1, ..., n \\ & \sum_{i=1}^{n}\alpha_i Y_i = 0 \end{aligned} \tag{5}$$

Once the optimization model is build any new test instance X_t can be classified in terms of learned α as shown in Equation 6.

$$Y_t = sgn\left(\sum_{i=1}^{n}\alpha_i Y_i \langle U_{i,*}, U_{t,*}\rangle + W_0\right) \tag{6}$$

The dot product, found in the dual formulation, between the instance vectors appears both in the optimization function 5 and the classification function 6. Such a dot product can be replaced by the so called kernel functions [21]. Various kernel representations exists, however in this study, for the sake of clarity and generality, we are going to prove the concept of the method using polynomial kernels, defined in Equation 7, which are known to be successful off-the-shelf kernels [21].

$$K(U_{i,*}, U_{l,*}) = \left(\sum_{k=1}^{d} U_{i,k}U_{l,k} + 1\right)^p \tag{7}$$

The ultimate objective function that defines nonlinear supervised dimensionality reduction is presented in Equation 8. *This model, in cooperation with the forthcoming learning algorithm, are the main contributions of our paper.*

$$\begin{aligned}\underset{U,V,\alpha,B_U,B_V}{\operatorname{argmin}} \quad F(X,Y,U,V,\alpha) = \beta \sum_{i=1}^{n+n'} \sum_{j=1}^{m} \left(X_{i,j} - \left(\sum_{k=1}^{d} U_{i,k} V_{k,j} + B_{U_i} + B_{V_j} \right) \right)^2 \\ + (1-\beta) \left(\frac{1}{2} \sum_{i=1}^{n} \sum_{l=1}^{n} \alpha_i \alpha_l Y_i Y_l \; K(U_{i,*}, U_{l,*}) - \sum_{i=1}^{n} \alpha_i \right) \\ + \lambda_U \sum_{i=1}^{n+n'} \sum_{k=1}^{d} U_{i,k}^2 + \lambda_V \sum_{k=1}^{d} \sum_{j=1}^{m} V_{k,j}^2 \end{aligned} \tag{8}$$

$$\begin{aligned} \text{s.t:} \quad & 0 \leq \alpha_i \leq C, \;\; i = 1, ..., n \\ & \sum_{i=1}^{n} \alpha_i Y_i = 0 \end{aligned}$$

Meanwhile the classification of a test instance U_t using kernels and the learned U, α, resulting from the solution of the dual joint optimization is shown in Equation 9.

$$Y_t = sgn \left(\sum_{i=1}^{n} \alpha_i Y_i \, K(U_{i,*}, U_{t,*}) + W_0 \right) \tag{9}$$

4 Learning Algorithm via Coordinate Descent

The objective function of Equation 8 is a non-convex function in terms of U, V and W, which makes it challenging for optimization. However stochastic gradient descent is shown to perform efficiently in minimizing such non-convex functions [12]. The benefits of stochastic gradient descent relies on better convergence, because cells of X are randomly picked for optimization, thus updating different rows of U, instead of iterating through the all the features of the same instance, thus resulting in subsequent updates of the same latent row of U.

On the other side, the classification accuracy terms of Equation 5 can be solved, in terms of α, by any standard SVMs dual solver method in case we consider U to be fixed. Thus, in an alternating fashion we solve the α-s by keeping U fixed. Then in the next step we update U using the learned α-s and V matrix, by taking a step in the negative direction of the overall loss w.r.t U. The update of V is performed as last step. Those three steps can be repeated until convergence as shown in the Algorithm 2.

Before starting the description of the algorithm let us define the gradients to be used for updating our latent matrices. We can represent the reconstruction loss F_R as sum of smaller loss terms $F_{Ri,j}$, per each cell (i, j) of the original

dataset X. Such decomposition will later enable stochastic gradient descent to optimize for each small loss term stochastically/randomly.

$$F_R(X,U,V) = \sum_{i=1}^{n+n'} \sum_{j=1}^{m} F_R(X,U,V)_{i,j} \tag{10}$$

$$F_R(X,U,V)_{i,j} = \beta \left(X_{i,j} - \left(\sum_{k=1}^{d} U_{i,k} V_{k,j} + B_{U_i} + B_{V_j} \right) \right)^2 + \lambda_U \frac{1}{m} \sum_{k=1}^{d} U_{i,k}^2 + \lambda_V \frac{1}{n+n'} \sum_{k=1}^{d} V_{k,j}^2 \tag{11}$$

Gradients:

$$e_{i,j} = X_{i,j} - \sum_{k=1}^{d} U_{i,k} V_{k,j} - B_{U_i} - B_{V_j}$$

$$\frac{\partial F_R(X,U,V)_{i,j}}{\partial U_{i,k}} = -2\beta\, e_{i,j}\, V_{k,j} + 2\lambda_U \frac{1}{m} U_{i,k} \tag{12}$$

$$\frac{\partial F_R(X,U,V)_{i,j}}{\partial V_{k,j}} = -2\beta\, e_{i,j}\, U_{i,k} + 2\lambda_V \frac{1}{n+n'} V_{k,j} \tag{13}$$

$$\frac{\partial F_R(X,U,V)_{i,j}}{\partial B_{U_i}} = -2\beta\, e_{i,j} \tag{14}$$

$$\frac{\partial F_R(X,U,V)_{i,j}}{\partial B_{V_j}} = -2\beta\, e_{i,j} \tag{15}$$

Similarly, we can split up the classification accuracy loss term, F_{CA}, into smaller loss terms $F_{CA\,i,l}$, defined per each instance pair (i,l).

$$F_{CA}(Y,U,\alpha) = \sum_{i=1}^{n} \sum_{l=1}^{n} F_{CA}(Y,U,\alpha)_{i,l} \tag{16}$$

$$F_{CA}(Y,U,\alpha)_{i,l} = (1-\beta) \left(\frac{1}{2} \alpha_i \alpha_l Y_i Y_l\; K(U_{i,*}, U_{l,*}) - \frac{1}{n^2} \sum_{i=1}^{n} \alpha_i \right) \tag{17}$$

Gradients:

$$\frac{\partial F_{CA}(Y,U,\alpha)_{i,l}}{\partial U_{i,k}} = (1-\beta)\, \frac{1}{2} \alpha_i \alpha_l Y_i Y_l\; p \left(\sum_{k=1}^{d} U_{i,k} U_{l,k} + 1 \right)^{p-1} U_{l,k} \tag{18}$$

$$\frac{\partial F_{CA}(Y,U,\alpha)_{i,l}}{\partial U_{l,k}} = (1-\beta)\, \frac{1}{2} \alpha_i \alpha_l Y_i Y_l\; p \left(\sum_{k=1}^{d} U_{i,k} U_{l,k} + 1 \right)^{p-1} U_{i,k} \tag{19}$$

The updates of α-s is carried through an algorithm which is a reduced version of the Sequential Minimal Optimization (SMO) [22]. Since the dual form optimization function contains the constraint $\sum_{i=1}^{n} \alpha_i Y_i = 0$, then any update of an α_i will violate the constraint. Therefore SMO updates the α-s in pair, offering

Algorithm 1 UpdateAlphaPair

Input: First alpha index i, Second alpha index j
Output: Updated α and W_0

$(\alpha_i^{old}, \alpha_j^{old}) \leftarrow (\alpha_i, \alpha_j)$
Let $s \leftarrow Y_i Y_j$
$(L, H) \leftarrow \left(\max(0, \alpha_j^{old} + s\alpha_i^{old} - \frac{s+1}{2}C), \min(C, \alpha_j^{old} + s\alpha_i^{old} - \frac{s-1}{2}C)\right)$
$E_k \leftarrow \left(\sum_{l=0}^{n} Y_l \alpha_l K(U_{l,*}, U_{k,*}) + W_0\right) - Y_k, \forall k \in \{i, j\}$
$\alpha_j^{new} \leftarrow \alpha_j^{old} - \frac{Y_j(E_i - E_j)}{2K(U_{i,*}, U_{j,*}) - K(U_{i,*}, U_{i,*}) - K(U_{j,*}, U_{j,*})^1}$

$$\alpha_j^{new,clipped} = \begin{cases} L\,, & \text{if } \alpha_j^{new} < L \\ \alpha_j^{new}\,, & \text{if } L < \alpha_j^{new} < H \\ H, & \text{if } \alpha_j^{new} > H \end{cases}$$

$\alpha_i^{new} \leftarrow \alpha_i^{old} + s(\alpha_j^{new,clipped} - \alpha_j^{old})$
$b_i \leftarrow E_i + y_i(\alpha_i^{new} - \alpha_i^{old})K(U_{i,*}, U_{i,*}) + Y_2(\alpha_j^{new,clipped} - \alpha_j^{old})K(U_{i,*}, U_{j,*}) + W_0$
$b_j \leftarrow E_j + y_i(\alpha_i^{new} - \alpha_i^{old})K(U_{i,*}, U_{i,*}) + Y_2(\alpha_j^{new,clipped} - \alpha_j^{old})K(U_{i,*}, U_{j,*}) + W_0$
$\mathbf{W_0} \leftarrow \frac{\mathbf{b_i + b_j}}{\mathbf{2}}$, $(\alpha_\mathbf{j}, \alpha_\mathbf{i}) \leftarrow \left(\alpha_\mathbf{j}^{\mathbf{new,clipped}}, \alpha_\mathbf{i}^{\mathbf{new}}\right)$
return α, W_0

three heuristics which defines which subset of the pairs should be updates first, in order to speed up the algorithm.

In difference to the original algorithm, we have ignored the selection heuristic for the *alpha* pairs to update. The reason for omitting the heuristics is due to the fact that U instances are continuously updated/modified. For instance, let us consider an imaginary instance U_i far away from the decision boundary, which means $\alpha_i = 0$. However in the next iteration, the instance U_i might be updated and move close to the boundary, meaning that α_i becomes a candidate for being updated ($0 < \alpha_i \leq C$), opposite to the functioning of SMO heuristic that would have avoided updating the instance, alluding that α_i is still 0.

The alpha updates rely on solving the function analytically for a pair of α-s at a step, until no $\alpha_i, \forall i$, violates the KKT [22] conditions described in Equation 20.

$$\begin{aligned} \text{Let } \hat{Y}_i \;=\; sgn\left(\sum_{j=1}^{n} \alpha_j Y_j\, K(U_{j,*}, U_{i,*}) + W_0\right) \\ \alpha_i = 0 \rightarrow Y_i\hat{Y}_i \geq 1 \\ 0 < \alpha_i < C \rightarrow Y_i\hat{Y}_i = 1 \\ \alpha_i = C \rightarrow Y_i\hat{Y}_i \leq 1 \end{aligned} \tag{20}$$

Therefore the learning algorithm will update all the pairs of α-s in each iteration. The SMO-like update of each pair of alphas is shown in the Algorithm 4, with more details in [22]. Please note that the algorithm also updates the hyperplane intercept W_0, which is used for classification of latent instances.

Having defined the gradients for updating latent matrices U, V with respect to the optimization loss and also the update rules for α-s, we can derive a final

learning algorithm based on coordinate gradient descent. Algorithm 2 shows the learning algorithm in full terms. The updates of each cell of U, V, B_U, B_V, as response to the reconstruction loss F_R and the classification accuracy loss F_{CA}, are conducted in the negative direction of the gradients scaled by hyper-parameter learning rates η_R, η_{CA}. The convergence is guaranteed by selecting small values for the learning rates. The stopping criteria is when the final loss from Equation 8 reaches an optimum, meaning it doesn't get further minimized.

[!t]

Algorithm 2 Learning Algorithm

Input: Dataset matrix $X \in \mathbb{R}^{(n+n')\times m}$, Labels vector $Y \in \mathbb{R}^n$, **Parameters:** { Box constraint C, Optimization switch β, Latent dimensions d, Learning rates η_R, η_{CA}, Regularizations λ_U, λ_V, Kernel degree p }
Output: $U, V, B_U, B_V, \alpha, W_0$
  Initialize $U \in \mathbb{R}^{(n+n')\times d}$, $V \in \mathbb{R}^{d\times m}$, $B_U \in \mathbb{R}^{(n+n')\times 1}$, $B_V \in \mathbb{R}^{1\times m}$ randomly
  Initialize $\alpha \leftarrow \{0\}^n$, $W_0 \leftarrow 0$
  while F **not** reached an optimum **do**
    for $\forall(i,j,k) \in (\{1...(n+n')\}, \{1...m\}, \{1...d\})$ *in random order* **do**
      $U_{i,k} \leftarrow U_{i,k} - \eta_R \frac{\partial F_R(X,U,V)_{i,j}}{\partial U_{i,k}}$
      $V_{k,j} \leftarrow V_{k,j} - \eta_R \frac{\partial F_R(X,U,V)_{i,j}}{\partial V_{k,j}}$
      $B_{U_i} \leftarrow B_{U_i} - \eta_R \frac{\partial F_R(X,U,V)_{i,j}}{\partial B_{U_i}}$
      $B_{V_j} \leftarrow B_{V_j} - \eta_R \frac{\partial F_R(X,U,V)_{i,j}}{\partial B_{V_j}}$
    end for
    for $\forall(i,l,k) \in (\{1...n\}, \{1...n\}, \{1...d\})$ *in random order* **do**
      $U_{i,k} \leftarrow U_{i,k} - \eta_{CA} \frac{\partial F_{CA}(Y,U,\alpha)_{i,l}}{\partial U_{i,k}}$
      $U_{l,k} \leftarrow U_{l,k} - \eta_{CA} \frac{\partial F_{CA}(Y,U,\alpha)_{i,l}}{\partial U_{l,k}}$
    end for
    for $\forall i \in \{1 \dots n\}$ **do**
      if α_i violates KKT of Equation 20 **then**
        for $\forall j \in \{1 \dots n\}$ *in random order* **do**
          $(\alpha, W_0) \leftarrow$ UpdateAlphaPair(i,j), from Algorithm 4
        end for
      end if
    end for
  end while
  return $U, V, B_U, B_V, \alpha, W_0$

5 Experimental Results

In order to compare the classification accuracy of our method Nonlinearly Supervised Dimensionality Reduction (**NSDR**), we implemented and compared against two baselines:

- **PCA-SVMs:** Matching against the standard PCA dimensionality reduction and then SVMs classification will demonstrate the advantage of supervised decomposition against unsupervised decomposition (PCA).
- **SVMs:** Comparison against the default SVMs will provide insights on the advantages of dimensionality reduction.

The experiments were conducted using five folds cross validation, where the data was divided into five splits and each split was, in turn, the test and the other four the training data.

The hyper parameters of our method and the baselines was selected using a validation data split from the training data. The best grid-search combinations of hyper parameters that yielded the best accuracy was selected for being applied to the test split. The ranges of search for the NSDR method were $\lambda_U \in \{10^{-6}, 10^{-5}, \ldots, 10^{0}, 10^{1}\}, \lambda_V \in \{10^{-6}, 10^{-5}, \ldots, 10^{0}, 10^{1}\}, \eta_R \in \{10^{-4}, 10^{-3}\}, \eta_{CA} \in \{10^{-4}, 10^{-3}\}, d \in \{25\%, 50\%, 75\%, 100\%\}$ of $m, \beta \in \{0.1, 0.5, 0.9\}, C \in \{0.1, 1, 10\}, p \in \{1, 2, 3, 4\}$. For PCA-SVMs there is a variance parameter $var \in \{0.5, 0.7, 1.0\} \times 100\%$. The other SVMs parameters C, p for both PCA-SVMs and SVMs were searched in the same ranges as the ones reported for NSDR previously.

5.1 Results and Interpretation

For the sake of empirical verification we randomly selected five popular binary datasets from the UCI repository. The results of the hyper parameter search over the selected datasets are shown in Table 1.

Table 1: **Hyper-parameter Search Results**

DATASET	NSDR	PCA-SVMs	SVMs
breast_cancer_wisconsin	$\lambda_U = 10^{-5}; \lambda_U = 10^{-1}; \eta_R = 10^{-3}; \eta_{CA} = 10^{-4}; d = 9; \beta = 0.9; C = 10; p = 2$	$var = 1; C = 10; p = 2$	$C = 10$ $p = 2$
ionosphere	$\lambda_U = 10^{-6}; \lambda_U = 10^{-6}; \eta_R = 10^{-3}; \eta_{CA} = 10^{-3}; d = 17; \beta = 0.5; C = 1; p = 2$	$var = 1; C = 1; p = 3$	$C = 0.1$ $p = 2$
pi-diabetes	$\lambda_U = 10^{-2}; \lambda_U = 10^{-6}; \eta_R = 10^{-3}; \eta_{CA} = 10^{-4}; d = 8; \beta = 0.5; C = 0.1; p = 3$	$var = 1$ $C = 1; p = 3$	$C = 10$ $p = 3$
sonar	$\lambda_U = 10^{-2}; \lambda_U = 10^{-4}; \eta_R = 10^{-3}; \eta_{CA} = 10^{-4}; d = 30; \beta = 0.1; C = 1; p = 3$	$var = 0.7$ $C = 10; p = 2$	$C = 0.1$ $p = 3$
spect	$\lambda_U = 10^{-2}; \lambda_U = 10^{0}; \eta_R = 10^{-3}; \eta_{CA} = 10^{-3}; d = 11; \beta = 0.1; C = 1; p = 3$	$var = 1$ $C = 1; p = 3$	$C = 0.1$ $p = 2$

There is a strong message we can derive from the hyper parameters results of Table 1. In no case the winning kernel degree was found to be $p = 1$, pointing to the conclusion that in all the listed datasets, non-linear dimensionality reduction (i.e. kernel degree $p > 1$) is superior.

The accuracy results in terms of error ratios is presented in Table 2. The winning method is shown in **bold**. As we can observe our proposed method

outperforms the baselines in the majority of the datasets as shown in the wins row.

Table 2: **Classification Accuracy - Error Ratios**

DATASET	**NSDR**	**PCA-SVMs**	**SVMs**
breast_cancer_wisconsin	**0.070 ± 0.018**	0.082 ± 0.019	0.073 ± 0.021
ionosphere	***0.066 ± 0.008**	0.091 ± 0.010	0.140 ± 0.018
pi-diabetes	0.287 ± 0.023	0.280 ± 0.006	**0.274 ± 0.030**
sonar	**0.202 ± 0.053**	0.226 ± 0.129	0.226 ± 0.056
spect	**0.206 ± 0.002**	0.243 ± 0.103	**0.206 ± 0.002**
Wins (sig/n.sig)	**3.5 (1/2.5)**	0	1.5 (1/0.5)

NSDR improves the classification on the *ionosphere* dataset with a significant difference, denoted by $*$, while on the other datasets the gap to the second best is smaller. It is interesting to observe that in the cases of *breast_cancer_wisconsin, pi-diabetes* and *spect* the performance of SVMs is better than PCA-SVMs. This observation leads to a reasoning that those datasets are hardly compressible, therefore unsupervised dimensionality reduction PCA is outperformed. However, due to the added advantage of nonlinear supervision, NSDR recovers the disadvantage of PCA-SVMs and wins on *breast_cancer_wisconsin* and co-wins on *spect*, while loosing only in the *pi-diabetes* dataset.

6 Conclusions and Future Work

Throughout this study we presented a nonlinearly supervised dimensionality reduction technique, which jointly combined a joint optimization on reconstruction and classification accuracy. The reconstruction terms were expressed as matrix factorization decomposition of latent matrices, while the classification accuracy as a dual form kernel maximum margin classifier. The reduced dataset is learned via a coordinate descent algorithm which updates the reduced dimensionality dataset w.r.t to both loss terms simultaneously. Empirical results over binary datasets shows that the proposed method outperforms the selected baselines in the majority of the datasets. Having proven the concept on binary classification, we plan to extend the model for multi-class data as future work.

References

1. Samet, H.: Foundations of Multidimensional and Metric Data Structures (The Morgan Kaufmann Series in Computer Graphics and Geometric Modeling). Morgan Kaufmann Publishers Inc., San Francisco, CA, USA (2005)
2. Grabocka, J., Nanopoulos, A., Schmidt-Thieme, L.: Classification of sparse time series via supervised matrix factorization. In Hoffmann, J., Selman, B., eds.: AAAI, AAAI Press (2012)
3. Das Gupta, M., Xiao, J.: Non-negative matrix factorization as a feature selection tool for maximum margin classifiers. In: Proceedings of the 2011 IEEE Conference

on Computer Vision and Pattern Recognition. CVPR '11, Washington, DC, USA, IEEE Computer Society (2011) 2841–2848
4. Jolliffe, I.T.: Principal Component Analysis. Second edn. Springer (October 2002)
5. Wismüller, A., Verleysen, M., Aupetit, M., Lee, J.A.: Recent advances in nonlinear dimensionality reduction, manifold and topological learning. In: ESANN. (2010)
6. Hoffmann, H.: Kernel pca for novelty detection. Pattern Recogn. **40**(3) (March 2007) 863–874
7. Sun, J., Crowe, M., Fyfe, C.: Extending metric multidimensional scaling with bregman divergences. Pattern Recognition **44**(5) (2011) 1137 – 1154
8. Gorban, A.N., Zinovyev, A.Y.: Principal manifolds and graphs in practice: from molecular biology to dynamical systems. Int. J. Neural Syst. **20**(3) (2010) 219–232
9. Lee, J.A., Verleysen, M.: Nonlinear dimensionality reduction. Springer, New York; London (2007)
10. Gashler, M.S., Martinez, T.: Temporal nonlinear dimensionality reduction. In: Proceedings of the IEEE International Joint Conference on Neural Networks IJCNN'11. IEEE Press (2011) 1959–1966
11. Singh, A.P., Gordon, G.J.: A unified view of matrix factorization models. In: ECML/PKDD (2). (2008) 358–373
12. Koren, Y., Bell, R.M., Volinsky, C.: Matrix factorization techniques for recommender systems. IEEE Computer **42**(8) (2009) 30–37
13. Rendle, S., Schmidt-Thieme, L.: Online-updating regularized kernel matrix factorization models for large-scale recommender systems. In Pu, P., Bridge, D.G., Mobasher, B., Ricci, F., eds.: RecSys, ACM (2008) 251–258
14. Cai, D., He, X., Han, J., Huang, T.S.: Graph regularized nonnegative matrix factorization for data representation. IEEE Trans. Pattern Anal. Mach. Intell. **33**(8) (2011) 1548–1560
15. Giannakopoulos, T., Petridis, S.: Fisher linear semi-discriminant analysis for speaker diarization. IEEE Transactions on Audio, Speech & Language Processing **20**(7) (2012) 1913–1922
16. Menon, A.K., Elkan, C.: Predicting labels for dyadic data. Data Min. Knowl. Discov. **21**(2) (2010) 327–343
17. Rish, I., Grabarnik, G., Cecchi, G., Pereira, F., Gordon, G.J.: Closed-form supervised dimensionality reduction with generalized linear models. In: ICML '08: Proceedings of the 25th international conference on Machine learning, New York, NY, USA, ACM (2008) 832–839
18. Fukumizu, K., Bach, F.R., Jordan, M.I.: Dimensionality reduction for supervised learning with reproducing kernel hilbert spaces. Journal of Machine Learning Research **5** (2004) 73–99
19. Salakhutdinov, R., Hinton, G.: Learning a nonlinear embedding by preserving class neighbourhood structure. In: Proceedings of the International Conference on Artificial Intelligence and Statistics. Volume 11. (2007)
20. Zhang, D., hua Zhou Songcan Chen, Z.: Semi-supervised dimensionality reduction. In: In: Proceedings of the 7th SIAM International Conference on Data Mining. (2007) 11–393
21. Scholkopf, B., Smola, A.J.: Learning with Kernels: Support Vector Machines, Regularization, Optimization, and Beyond. MIT Press, Cambridge, MA, USA (2001)
22. Platt, J.: Fast training of support vector machines using sequential minimal optimization. In Schoelkopf, B., Burges, C., Smola, A., eds.: Advances in Kernel Methods - Support Vector Learning. MIT Press (1998)