Abstract

Feature selection plays a fundamental role in many pattern recognition problems. However, most efforts have been focused on the supervised scenario, while unsupervised feature selection remains as a rarely touched research topic. In this paper, we propose Manifold-Based Maximum Margin Feature Selection (M3FS) to select the most discriminative features for clustering. M3FS targets to find those features that would result in the maximal separation of different clusters and incorporates manifold information by enforcing smoothness constraint on the clustering function. Specifically, we define scale factor for each feature to measure its relevance to clustering, and irrelevant features are identified by assigning zero weights. Feature selection is then achieved by the sparsity constraints on scale factors. Computationally, M3FS is formulated as an integer programming problem and we propose a cutting plane algorithm to efficiently solve it. Experimental results on both toy and real-world data sets demonstrate its effectiveness.

1. Introduction

Real-world data sets are often high-dimensional and contain many spurious features. For example, in face recognition, an image of size $m \times n$ is often represented as a vector in $\mathbb{R}^{mn}$, which can be very high-dimensional for typical values of $m$ and $n$. Similarly, biological databases such as microarray data can have thousands or even tens of thousands of genes as features. Such a large number of features can easily lead to the curse of dimensionality and severe over-fitting. Hence, dimensionality reduction, in the form of either feature extraction or feature selection, plays a fundamental role in many pattern recognition problems.

In this paper, we will focus on feature selection, which selects a relevant subset of features. Excellent reviews on this topic can be found in \[8, 10\]. Note that not only can feature selection improve the generalization performance of the resultant classifier, the use of fewer features is also less computationally expensive and thus implies faster testing. Moreover, it can eliminate the need to collect a large number of irrelevant and redundant features, and thus reduces the cost. Finally, with the discovery of fewer features, the resultant model can be more easily understood by human.

In feature selection, the features may be scored either individually or as a subset. In general, there are three approaches to score them: filters, wrappers, and embedded methods \[8\]. Filters score the features as a pre-processing step, independently of the classifier. Wrappers score the features according to their prediction performance when used with the classifier. Both filters and wrappers rely on search strategies to guide the search for the “best” feature subset. While a large number of search strategies can be used, often one is limited to the computationally simple greedy (forward or backward) strategies. Finally, embedded methods combine feature selection with the classifier. While the design of embedded methods is tightly coupled with the specific classifier, they are often considered as more efficient than filters and wrappers \[8\].

While supervised feature selection has been extensively studied for decades, feature selection in the unsupervised learning setting has received relatively little attention. This is partly due to the fact that unsupervised feature selection is much more difficult because of the lack of label information to guide the search for relevant features. While most unsupervised feature selection methods are based on the filter approach \[6, 12, 14\], some wrappers \[16\] and embedded approaches that treat clustering and feature selection simultaneously have also been proposed \[4, 7, 12\]. However, these are often based on generative models (such as Gaussian mixtures) \[4, 7, 12, 16\]. As is well-known, generative models may lead to inferior performance when the model assumption does not match the observed data.

Instead of relying on model-based clustering, we will propose in this paper an embedded method that is based on discriminative clustering. This is motivated by the common belief that discriminative models are often better than gen-
erative models in supervised learning. Among the discriminative methods, large margin methods, such as the support vector machines, are particularly successful. Indeed, inspired by the superiority of large margin methods in supervised learning, there is growing interest in extending them to unsupervised learning. For example, Xu et al. [21] proposed a novel approach called maximum margin clustering (MMC), which performs clustering by simultaneously finding the large margin separating hyperplane between clusters. Experimental results showed that this large margin clustering method (and its variant [19]) have been very successful in many clustering problems.

Moreover, in many computer vision and pattern recognition applications (such as face recognition and hand-written digit recognition), it has been observed that the data examples often lie on a manifold. Hence, another novelty of the proposed approach is that manifold information can also be incorporated into the feature selection process. Note that the Laplacian score [9], which can be used as a filter approach for unsupervised feature selection, also utilizes manifold information. However, for the Laplacian score, a feature will be considered as good if two samples that are close to each other on the data manifold are also close to each other according to that feature. On the other hand, the proposed method uses the manifold information by directly considering the resultant decision function and ensures that it is smooth on the manifold. As will be seen in Section 4, since ours is an embedded method that explicitly considers the clustering objective, it performs much better than the filter method of Laplacian score.

In this paper, we propose Manifold-Based Maximum Margin Feature Selection (M3FS) to select the most discriminative features for clustering. M3FS targets to find those features that would result in the maximal separation of different clusters and incorporates manifold information by enforcing smoothness constraint on the clustering function. Specifically, we define a scale factor for each feature to measure its relevance to clustering, and irrelevant features are identified by assigning zero weights. Feature selection is then achieved by the sparsity constraints on the scale factors. Computationally, M3FS is formulated as an integer programming problem and we propose a cutting plane algorithm to efficiently solve it. Experimental results on both toy and real-world data sets demonstrate its effectiveness.

The rest of this paper is organized as follows. In Section 2, we present a brief introduction to maximum margin clustering. Section 3 presents the details of the M3FS algorithm, together with theoretical analysis on both the accuracy and time complexity of the algorithm, and extension to the multi-class clustering setting. Experimental results on both toy and real-world data sets are provided in Section 4, followed by some concluding remarks in Section 5.

2. Maximum Margin Clustering

Maximum margin clustering (MMC) is a recently proposed clustering algorithm that extends support vector machines (SVM) to unsupervised learning setting. Since the class labels are unknown in unsupervised learning, MMC tries to find a cluster labeling of the patterns, together with a hyperplane classifier, such that the resultant margin is maximized among all possible labelings [21].

For simplicity of exposition, assume that there are only two clusters. Given a set of examples \( X = \{x_1, \ldots, x_n\} \in \mathbb{R}^{d \times n} \), MMC targets to find the best label combination \( y = \{y_1, \ldots, y_n\} \in \mathbb{R}^n \in \{-1, +1\}^n \) such that an SVM trained on this \( \{(x_1, y_i), \ldots, (x_n, y_i)\} \) yields the largest margin. Computationally, it can be formulated as the following problem

\[
\min_{w, b, \xi} \frac{1}{2} w^T w + \frac{C}{n} \sum_{i=1}^{n} \xi_i \\
\text{s.t.} \ \\
\forall i \in \{1, \ldots, n\} : \\
y_i(w^T x_i + b) \geq 1 - \xi_i, \ \xi_i \geq 0, \\
-l \leq \sum_{i=1}^{n} y_i \leq l.
\]

where \( \sum_{i=1}^{n} \xi_i \) is divided by \( n \) to better capture how \( C \) scales with the data set size. The last constraint in (1) is often known as the class balance constraint. It is introduced to avoid the trivially “optimal” solution that assigns all patterns to the same class and thus achieves “infinite” margin. Here, \( l > 0 \) is a constant controlling the class imbalance.

3. Maximum Margin Feature Selection with Manifold Regularization

In this section, we present the manifold-based maximum margin feature selection algorithm. We will first consider the two-cluster case. Extension to the multi-class case will be discussed in Section 3.5.

3.1. Two-Class Manifold-Based Maximum Margin Feature Selection

Manifold-based maximum margin feature selection (M3FS) is an embedded approach that performs clustering and feature selection simultaneously. It tries to find a subset of the \( d \) given features such that the resultant clusters will be maximally separated. As mentioned in Section 1, while previous efforts on unsupervised feature selection are often based on generative models which require strong model assumption, M3FS adopts maximum margin clustering which can often outperform conventional clustering methods.

Moreover, in many computer vision and pattern recognition applications, it has been observed that the data exam-
amples often lie on a manifold. Hence, our goal is to also utilize this manifold information in the feature selection process. As is well-known, the data manifold can be represented by a graph \( \mathcal{G} \). In the following, let \( W \in \mathbb{R}^{n \times n} \) be the similarity (or adjacency) matrix of \( \mathcal{G} \), \( D \in \mathbb{R}^{n \times n} \) be the diagonal degree matrix whose \( i \)-th entry is the sum of the \( i \)-th row of \( W \), and \( L = I - D^{-\frac{1}{2}}WD^{-\frac{1}{2}} \) (where \( I \) is the \( n \times n \) identity matrix) be the normalized graph Laplacian [3].

To achieve the first goal, we extend MMC by associating each feature \( k \) \((k = 1, 2, \ldots, d)\) with a learnable scale factor \( \sigma_k \), which is used to measure its “relevance” to clustering. When learning is completed, the irrelevant features can then be identified as those having zero scale factors [20]. Hence, \( \sigma \) can thus be formulated as the following optimization problem:

\[
\min_{\sigma, b, \xi, \mathbf{w}} \frac{1}{2} \sum_{k=1}^{d} \sigma_k w_k^2 + \sum_{i=1}^{n} \xi_i + \lambda |X^T \mathbf{w} + b|^T L |X^T \mathbf{w} + b| \\
\text{s.t. } \forall i \in \{1, \ldots, n\} : \xi_i \geq 0,
\]

\[
y_i \left( \sum_{k=1}^{d} \sigma_k w_k x_{ik} + b \right) \geq 1 - \xi_i,
\]

\[
\forall k \in \{1, \ldots, d\} : 0 \leq \sigma_k \leq 1; y \in \{-1, +1\}^n
\]

\[
\sum_{k=1}^{d} \sigma_k = m,
\]

\[
-l \leq \sum_{i=1}^{n} \left( \sum_{k=1}^{d} \sigma_k w_k x_{ik} + b \right) \leq l
\]

where \( \lambda \) is a user-defined regularization parameter, and \( m \) is the number of features to be selected. Note that we have also relaxed the constraint \( \sigma_k \in \{0, 1\} \) on \( \sigma \) to \( 0 \leq \sigma_k \leq 1 \). The \( \ell_1 \) regularizer (4) on \( \sigma \) enforces sparsity. Moreover, a slightly relaxed class balance constraint is used in (5) [17].

Since \( \sigma_k \) and \( w_k \) are coupled together in the decision function, the objective in (2) and the constraints (3), (5) are non-convex. Therefore, we apply the change of variables [24]: \( \forall k \in \{1, \ldots, d\} : v_k = \sigma_k w_k \). Let \( v = [v_1, v_2, \ldots, v_d]^T \), we have the following proposition:

**Proposition 1** M3FS can be equivalently formulated as

\[
\min_{v, b, \xi, \mathbf{w}} \frac{1}{2} \sum_{k=1}^{d} v_k^2 + C \sum_{i=1}^{n} \xi_i + \lambda (X^T v + b)^T L (X^T v + b) \\
\text{s.t. } \forall i \in \{1, \ldots, n\} : |v^T x_i + b| \geq 1 - \xi_i,
\]

\[
\forall k \in \{1, \ldots, d\} : 0 \leq \sigma_k \leq 1,
\]

\[
\sum_{k=1}^{d} \sigma_k = m,
\]

\[
l \leq \sum_{i=1}^{n} (v^T x_i + b) \leq l
\]

where \( y \) is calculated as \( y_i = \text{sgn}(v^T x_i + b) \).

### 3.2. Cutting Plane Algorithm

The M3FS formulation in (6) has \( n \) slack variables \( \xi_i \)'s, one for each data sample \( x_i \). We reformulate (6) as follows to reduce the number of slack variables,

\[
\min_{v, b, \xi, \mathbf{w}} \frac{1}{2} \sum_{k=1}^{d} v_k^2 + C \sum_{i=1}^{n} \xi_i + \lambda (X^T v + b)^T L (X^T v + b) \\
\text{s.t. } \forall i \in \{0, 1\}^n : \frac{1}{n} \sum_{i=1}^{n} c_i |v^T x_i + b| \geq \frac{1}{n} \sum_{i=1}^{n} c_i - \xi_i,
\]

\[
\forall k \in \{1, \ldots, d\} : 0 \leq \sigma_k \leq 1,
\]

\[
\sum_{k=1}^{d} \sigma_k = m,
\]

\[
l \leq \sum_{i=1}^{n} (v^T x_i + b) \leq l
\]

**Proposition 2** Any solution \((v^*, b^*, \xi^*, \sigma^*)\) to problem (7) is also a solution to problem (6), and vice versa, with \( \xi^* = \frac{1}{n} \sum_{i=1}^{n} \xi_i^* \).

The number of slack variables is now reduced by \( n - 1 \). On the other hand, the number of constraints in (7) is increased from \( n \) to \( 2^n \). To handle this exponential number of constraints, we employ an adaptation of the cutting plane algorithm [11]. It starts with an empty constraint subset \( \Omega \), and computes the optimal solution to problem (7) subject to the constraints in \( \Omega \). The algorithm then finds the most violated constraint in (8) and adds it to \( \Omega \). In this way, we construct a series of successively tightening approximations to problem (7). The algorithm stops when no constraint in (8) is violated by more than \( \epsilon \). The whole cutting plane algorithm for M3FS is presented in Algorithm 1.

#### 3.2.2 Optimization via the CCCP

For the optimization problem in (7), the objective is convex (quadratic) and all the constraints except the first one are linear. Moreover, note that although the constraint in (8) is
computes $b$ following semi-definite, the above problem can be reformulated as non-convex, it can be expressed as a difference of the two convex functions $\frac{1}{n} \sum_{i=1}^{n} c_i |v^T x_i + b|$ and $\frac{1}{n} \sum_{i=1}^{n} c_i - \xi$. Hence, we can solve problem (7) with the constrained concave-convex procedure (CCCP), which is designed to solve these optimization problems with a concave-convex objective function and concave-convex constraints [18]. Specifically, given an initial estimate $(v^{(0)}, b^{(0)})$, the CCCP computes $(v^{(l+1)}, b^{(l+1)})$ from $(v^{(l)}, b^{(l)})$ by replacing $\frac{1}{n} \sum_{i=1}^{n} c_i |v^T x_i + b|$ in the first constraint with its first-order Taylor expansion at $(v^{(l)}, b^{(l)})$, leading to

$$\min_v \frac{1}{2} \sum_{k=1}^{d} v_k^2 + C^T \xi + \lambda (X^T v + b)^T L (X^T v + b) \tag{9}$$

s.t. $\forall c \in \Omega : \frac{1}{n} \sum_{i=1}^{n} c_i z_i^{(l)} (v^T x_i + b) \geq \frac{1}{n} \sum_{i=1}^{n} c_i - \xi$, $\forall k \in \{1, \ldots, d\} : 0 \leq \sigma_k \leq 1$, $\sum_{k=1}^{d} \sigma_k = m$, $\xi \geq 0$, $-l \leq \sum_{i=1}^{n} (v^T x_i + b) \leq l$

where $z_i^{(l)} = \text{sgn}(v^{(l)} x_i + b)$. Define $t_k$ as the upper bound of $s^2$, $s$ as the upper bound of $(X^T v + bL (X^T v + b))$, and note that $L$ is symmetric positive semi-definite, the above problem can be reformulated as the following second order cone programming (SOCP) [2].

$$\min_v \frac{1}{2} \sum_{k=1}^{d} t_k + C^T \xi + \lambda s \tag{10}$$

s.t. $\forall c \in \Omega : \frac{1}{n} \sum_{i=1}^{n} c_i z_i^{(l)} (v^T x_i + b) \geq \frac{1}{n} \sum_{i=1}^{n} c_i - \xi$, $\forall k \in \{1, \ldots, d\} : 0 \leq \sigma_k \leq 1$, $\forall k \in \{1, \ldots, d\} : \left\| \frac{2v_k t_k - \sigma_k}{s - 1} \right\| \leq t_k + \sigma_k$, $\left\| \frac{2L x}{s - 1} \right\| \leq s + 1$, $\xi \geq 0$, $\sum_{k=1}^{d} \sigma_k = m$, $-l \leq \sum_{i=1}^{n} (v^T x_i + b) \leq l$

The above SOCP problem can be solved in polynomial time [13]. Following the CCCP, the obtained solution $(v, b, \sigma, \xi, t, s)$ from this SOCP problem is then used as $(v^{(l+1)}, b^{(l+1)}, \sigma, \xi, t, s)$, and the iteration continues until convergence. The algorithm for solving problem (7) subject to the constraint subset $\Omega$ is summarized in Algorithm 2. As for its termination criterion, we check if the difference in objective values from two successive iterations is less than $\alpha\%$ (which is set to 0.01 in the experiments).

**Algorithm 1** Cutting plane algorithm for M3FS

**Input:** $X$, $C$, $l$, $\lambda$ and $\epsilon$, set constraint subset $\Omega = \emptyset$.

**repeat**

Solve problem (7) for $(v, b, \sigma, \xi)$ under the current working constraint set $\Omega$.

Select the most violated constraint $c$; set $\Omega = \Omega \cup \{c\}$.

**until** the newly selected constraint $c$ is violated by no more than $\epsilon$.

The cutting plane algorithm iteratively selects the most violated constraint under the current hyperplane parameter and then adds it to the working constraint set $\Omega$, until no constraint is violated by more than $\epsilon$, i.e.,

$$\forall c \in \{0, 1\}^n : \frac{1}{n} \sum_{i=1}^{n} c_i |v^T x_i + b| \geq \frac{1}{n} \sum_{i=1}^{n} c_i - (\xi + \epsilon) \tag{12}$$

Moreover, note that in the objective function of problem (7), there is a single slack variable $\xi$ measuring the clustering loss. Hence, we can simply select the stopping criterion in Algorithm 1 as being all the samples satisfying inequality (12). Then, the approximation accuracy $\epsilon$ of this approximate solution is directly related to the clustering loss.

**3.2.2 Identifying the Most Violated Constraint**

The most violated constraint is the one that results in the largest $\xi$. Since each constraint in (8) is represented by a vector $c$, we have the following proposition:

**Proposition 3** The most violated constraint $c$ in (8) can be computed as:

$$c_i = \begin{cases} 1 & \text{if } |v^T x_i + b| < 1, \\ 0 & \text{otherwise} \end{cases} \tag{11}$$

The cutting plane algorithm iteratively selects the most violated constraint under the current hyperplane parameter and then adds it to the working constraint set $\Omega$, until no constraint is violated by more than $\epsilon$, i.e.,

$\forall c \in \{0, 1\}^n : \frac{1}{n} \sum_{i=1}^{n} c_i |v^T x_i + b| \geq \frac{1}{n} \sum_{i=1}^{n} c_i - (\xi + \epsilon) \tag{12}$

Moreover, note that in the objective function of problem (7), there is a single slack variable $\xi$ measuring the clustering loss. Hence, we can simply select the stopping criterion in Algorithm 1 as being all the samples satisfying inequality (12). Then, the approximation accuracy $\epsilon$ of this approximate solution is directly related to the clustering loss.

**3.3 Accuracy of the Cutting Plane Algorithm**

The following proposition characterizes the accuracy of the solution computed by the cutting plane algorithm.

**Proposition 4** For any $\epsilon > 0$, the cutting plane algorithm for M3FS returns a point $(v, b, \sigma, \xi)$ for which $(v, b, \sigma, \xi + \epsilon)$ is feasible in problem (7).

Based on this proposition, $\epsilon$ indicates how close one wants to be to the error rate of the best separating hyperplane. This justifies its use as the stopping criterion in Algorithm 1.
3.4. Time Complexity Analysis

In this section, we provide theoretical analysis on the time complexity of the cutting plane algorithm for manifold-based maximum margin feature selection. We will first obtain the time involved in each iteration of the algorithm. Next, we will show that the total number of constraints added into the working set \( \Omega \), i.e., the total number of iterations involved in the cutting plane algorithm, is upper bounded. Specifically, we have the following two lemmas.

**Lemma 1** Each iteration of the cutting plane algorithm for manifold-based maximum margin feature selection takes \( O(d^{3.5} + nd + d^{2.5} |\Omega|) \) time for a working constraint set size \(|\Omega|\).

**Lemma 2** The cutting plane algorithm terminates after adding at most \( \frac{CR}{\epsilon^2} \) constraints, where \( R \) is a constant independent of \( n \) and \( d \).

Lemma 2 bounds the number of iterations in our cutting plane algorithm by a constant \( \frac{CR}{\epsilon^2} \), which is independent of \( n \) and \( d \). Moreover, each iteration of the algorithm takes \( O(d^{3.5} + nd + d^{2.5} |\Omega|) \) time. Therefore, the cutting plane algorithm for manifold-based maximum margin feature selection has a time complexity of \( \sum_{|\Omega|=1}^{CR/\epsilon^2} O(d^{3.5} + nd + d^{2.5} |\Omega|) = O(\frac{d^{3.5} + nd + d^{2.5}}{\epsilon^2}) \). Hence, we have the following proposition.

**Proposition 5** The cutting plane algorithm for manifold-based maximum margin feature selection takes \( O(\frac{d^{3.5} + nd + d^{2.5}}{\epsilon^2}) \) time.

3.5. Multi-Class M3FS

For the multi-class scenario, we will start with an introduction to the multi-class support vector machine formulation proposed in [5]. Given a point set \( X = \{x_1, \ldots, x_n\} \) and their labels \( y = (y_1, \ldots, y_n) \in \{1, \ldots, M\}^n \), the SVM defines a weight vector \( w_p \) for each class \( p \in \{1, \ldots, M\} \) and classifies sample \( x \) by \( p^* = \arg \max_{p \in \{1, \ldots, M\}} w_p^T x \). The weight vectors are obtained as follows:

\[
\min_{w_1, \ldots, w_M, \xi} \; \frac{1}{2} \sum_{p=1}^{M} ||w_p||^2 + C \sum_{i=1}^{n} \xi_i \quad \text{(13)}
\]

s.t. \( \forall i \in \{1, \ldots, n\}, \; r \in \{1, \ldots, M\} : \)
\[
w_{y_i}^T x_i + \delta_{y_i, r} - w_r^T x_i \geq 1 - \xi_i; \; \xi_i \geq 0.
\]

Similar with the two-class scenario, we define a scale factor for each feature and obtain the following unsupervised multi-class manifold-based maximum margin feature selection formulation

\[
\min_{y, \sigma, v, \xi} \; \frac{1}{2} \sum_{k=1}^{d} \sum_{p=1}^{M} \sigma_k^2 + C \sum_{i=1}^{n} \xi_i + \lambda \sum_{p=1}^{M} v_p^T X L X^T v_p \quad \text{(14)}
\]

s.t. \( \forall i \in \{1, \ldots, n\}, \; r \in \{1, \ldots, M\} : \)
\[
\sum_{k=1}^{d} (v_{y_i, k} - v_{r, k}) x_{i, k} + \delta_{y_i, r} \geq 1 - \xi_i, \; \xi_i \geq 0 \]
\[
\forall k \in \{1, \ldots, d\} : \; 0 \leq \sigma_k \leq 1; \sum_{k=1}^{d} \sigma_k = m \]
\[
\forall p, q \in \{1, \ldots, M\} : -l \leq \sum_{i=1}^{n} (v_{pk} - v_{qk}) x_{i, k} \leq l,
\]

Here, the subscript \( p \) in \( w_{pk} \) denotes the \( p \)th class, \( k \) denotes the \( k \)th feature, and we have applied the change of variables \( \forall p \in \{1, \ldots, M\}, \; k \in \{1, \ldots, d\} : \; v_{pk} = \sigma_k w_{pk} \) to ensure that the objective function and the last constraint are convex. Similar to two-class clustering, we have also added class balance constraints (where \( l > 0 \)) in the formulation to control class imbalance. Again, the above formulation is an integer program, and is much more complex than the QP problem in multi-class SVM. Fortunately, we have the following proposition.

**Proposition 6** Problem (14) is equivalent to

\[
\min_{\sigma, v, \xi} \; \frac{1}{2} \sum_{k=1}^{d} \sum_{p=1}^{M} \sigma_k^2 + C \sum_{i=1}^{n} \xi_i + \lambda \sum_{p=1}^{M} v_p^T X L X^T v_p \quad \text{(15)}
\]

s.t. \( \forall i \in \{1, \ldots, n\}, \; r \in \{1, \ldots, M\} : \)
\[
\sum_{k=1}^{d} (z_{ip} v_{pk} - v_{rk}) x_{i, k} + z_{ir} \geq 1 - \xi_i, \; \xi_i \geq 0 \]
\[
\forall k \in \{1, \ldots, d\} : \; 0 \leq \sigma_k \leq 1; \sum_{k=1}^{d} \sigma_k = m \]
\[
\forall p, q \in \{1, \ldots, M\} : -l \leq \sum_{i=1}^{n} (v_{pk} - v_{qk}) x_{i, k} \leq l,
\]

where \( z_{ip} \) is defined as \( \forall i \in \{1, \ldots, n\}, \; p \in \{1, \ldots, M\} : \)
\[
z_{ip} = \prod_{q=1}^{M} I_{[\sum_{k=1}^{d} v_{pk} x_{i, k} > \sum_{k=1}^{d} v_{qk} x_{i, k}]},
\]

with \( I(\cdot) \) being the indicator function and the label for sample \( x_i \) is determined as \( y_i = \arg \max_p \sum_{k=1}^{d} v_{pk} x_{i, k} = \sum_{p=1}^{M} \prod_{q \neq p} z_{ip} \).

To reduce the number of slack variables, we make use of the following proposition:
Proposition 7 Problem (15) can be equivalently formulated as problem (16), with $\xi^* = \frac{1}{d}\sum_{k=1}^{M} \xi_k^*$:

$$\begin{align*}
\min_{\sigma, \xi} & \quad \frac{1}{2} \sum_{k=1}^{M} \sum_{p=1}^{n} z_{kp}^2 + C\xi + \lambda \sum_{p=1}^{M} v_p^T XLX^T v_p \\
\text{s.t.} & \quad \forall k \in \{1, \ldots, M\} : \frac{1}{n} \sum_{i=1}^{n} \sum_{p=1}^{M} (c_i^T e_{z_{kp} - c_{ip}}) v_{pk} x_{ik} + \frac{1}{n} \sum_{i=1}^{n} \sum_{p=1}^{M} c_{ip} z_{ip} \\
& \quad \geq \frac{1}{n} \sum_{i=1}^{n} c_i^T e - \xi, \\
& \quad \forall k \in \{1, \ldots, d\} : 0 \leq \sigma_k \leq 1,
\end{align*}$$

(16)

Proposition 8 The most violated constraint $c = [c_1, \ldots, c_n]$ can be obtained as:

$$c_i = \begin{cases} e \ast_p & \text{if } \left[\sum_{k=1}^{d} v_{pk} x_{ik} - \sum_{k=1}^{d} v_{rk} x_{ik}\right] < 1, \\
0 & \text{otherwise},
\end{cases}$$

where $p = \arg \max_p \sum_{k=1}^{d} v_{pk} x_{ik}$ and $r^* = \arg \max_{r \neq p^*} \sum_{k=1}^{d} v_{rk} x_{ik}$.

4. Experiments

In this section, we validate the effectiveness of manifold-based maximum margin feature selection (M3FS) on both toy and real-world data sets.

4.1. Setup

We use 5 data sets which are intended to cover a wide range of properties: ionosphere, digits, letter and satellite (these are from the UCI data repository\(^1\)), and mnist\(^2\).

The two-class data sets are created following the same setting as in [22]. We also create several multi-class data sets from the digits, letter and mnist data. All these are summarized in Table 1. For representing the manifold used in M3FS, we use a fully-connected graph connecting all the samples, and set the pairwise similarity matrix $W$ as $w_{ij} = \exp \left( -\frac{1}{2} \|x_i - x_j\|_2^2 / \rho^2 \right)$, where $\rho$ is the variance in the Gaussian function. Besides M3FS, for comparison, we also run the following algorithms which perform clustering with feature selection:

- Feature selection based on Gaussian mixture model (FSGMM) [12]: This is an embedded approach for unsupervised feature selection and, as its name implies, the clustering algorithm is based on the Gaussian mixture model. Its implementation is the same as in [12].

- Laplacian score [9]: This is a filter method for supervised/unsupervised feature selection which also uses manifold information. The implementation code is downloaded from

\(^1\)http://archive.ics.uci.edu/ml/
\(^2\)http://yann.lecun.com/exdb/mnist/

<table>
<thead>
<tr>
<th>Data</th>
<th>Size</th>
<th>Feature</th>
<th>Class</th>
</tr>
</thead>
<tbody>
<tr>
<td>digits1v7</td>
<td>361</td>
<td>64</td>
<td>2</td>
</tr>
<tr>
<td>digits2v7</td>
<td>356</td>
<td>64</td>
<td>2</td>
</tr>
<tr>
<td>ionosphere</td>
<td>354</td>
<td>64</td>
<td>2</td>
</tr>
<tr>
<td>letterAvB</td>
<td>1555</td>
<td>16</td>
<td>2</td>
</tr>
<tr>
<td>satellite</td>
<td>2236</td>
<td>36</td>
<td>2</td>
</tr>
<tr>
<td>digits0689</td>
<td>713</td>
<td>64</td>
<td>4</td>
</tr>
<tr>
<td>digits1279</td>
<td>718</td>
<td>64</td>
<td>4</td>
</tr>
<tr>
<td>letterABCD</td>
<td>3096</td>
<td>16</td>
<td>4</td>
</tr>
<tr>
<td>mnist01234</td>
<td>2891</td>
<td>196</td>
<td>5</td>
</tr>
</tbody>
</table>

Table 1. Descriptions of the data sets.
Moreover, we also experiment with maximum margin clustering without doing feature selection. The implementation is the same as in [23], and this will be denoted as MMC-all.

In the experiments, we first take a set of labeled data, remove all the labels and run the clustering algorithms; then we label each of the resulting clusters with the majority class according to the original labels. Moreover, we always set the number of clusters to be the true number of classes $M$ for all the methods. These clustering algorithms (with or without feature selection) will be evaluated by the following two performance measures:

**Clustering Accuracy (Acc).** The first performance measure is the Clustering Accuracy, which discovers the one-to-one relationship between clusters and classes and measures the extent to which each cluster contained data points from the corresponding class. Specifically, Acc measures the number of correct classifications.

**Rand Index (RI)** [15]. Let $C = \{C_1, C_2, \ldots, C_M\}$ be the set of final clustering results such that $C_k$ represents the $k$th cluster, and $L = \{L_1, L_2, \ldots, L_M\}$ denotes the set of true data classes such that $L_k$ represents the $k$th class. We define the following four variables: $a$: the number of data pairs in $X$ that are in the same set in both $C$ and $L$; $b$: the number of data pairs in $X$ that are in different sets in both $C$ and $L$; $c$: the number of data pairs in $X$ that are in the same set in $C$ but different sets in $L$; $d$: the number of data pairs in $X$ that are in different sets in $C$ but the same set in $L$. Then the Rand Index $R$ that measures the similarity between $C$ and $L$ can be computed as $R = \frac{a+b}{a+b+c+d}$. Intuitively, one can think of $a+b$ as the number of agreements between $C$ and $L$ and $c+d$ as the number of disagreements between $C$ and $L$. Clearly, $R$ has a value between 0 and 1, with 0 indicating that $C$ and $L$ do not agree on any pair of data points, and 1 indicating that $C$ and $L$ are exactly the same.

### 4.2. Ability to Detect Relevant Features

In this section, we first illustrate the ability of $M3FS$ in selecting relevant features by using the iris data set from UCI machine learning repository. The iris data contain 3 classes of 50 instances each, and each instance is characterized by 4 features. We add 10 noisy features (generated from the normal distribution $N(0, 1)$) to the iris data, and thus obtain a data set of 150 14-dimensional instances.

The saliencies of all the 14 features as calculated by the various methods are shown in Figure 1. For simplicity of illustration, we order the features such that the first four are the original features, while the last ten are the noisy ones. As can be seen, $M3FS$ successfully selects the 4 relevant features and assigns zero saliency to all the noisy features. On the other hand, both the Laplacian score and FSGMM assign non-zero saliencies to the noisy features.

![Figure 1. Feature saliencies on the iris data set with 10 noisy features added.](image)

### 4.3. Clustering Performance

In this section, we report the clustering performance of the various algorithms on the data sets in Table 1. The clustering accuracy and Rand Index results are shown in Table 2. We also demonstrate the effect when manifold information is not used by setting $\lambda = 0$. As can be seen, even when no manifold information is used, both the clustering accuracy and Rand Index of $M3FS$ are comparable to those attained by maximum margin clustering using all features and is often better than the other unsupervised feature selection algorithms. The addition of manifold regularization significantly improves the performance of $M3FS$ and enables it to be even better than MMC-all.

### 4.4. Generalization Ability of M3FS

**Manifold-based maximum margin feature selection** adopts the maximum margin principle of SVM, which could allow good generalization on unseen data. In this experiment, we validate the generalization ability of $M3FS$ on unseen data samples. We first learn the $M3FS$ model on a data subset randomly drawn from the whole data set. Then we use the learned model to cluster the whole data set. As can be seen in Table 3, the clustering performance of the model learned on the data subset is comparable with that of the model learned on the whole data set. Thus, for a large data set, we can simply perform the feature selection and clustering process on a small subset of the data and then use the learned model to cluster the remaining data points.

### 5. Conclusions

In this paper, we propose a novel unsupervised feature selection method named Manifold-Based Maximum Margin Feature Selection (M3FS). $M3FS$ targets to identify those features that would result in the maximal separation of different clusters. As many computer vision and pattern recognition problems have intrinsic manifold structure, we add
<table>
<thead>
<tr>
<th>Data</th>
<th>m</th>
<th>LapKM</th>
<th>LapMMC</th>
<th>FSGMM</th>
<th>MMC-all</th>
<th>M3FS</th>
<th>M3FS (λ = 0)</th>
</tr>
</thead>
<tbody>
<tr>
<td>digitsv17</td>
<td>10</td>
<td>79.50</td>
<td>0.569</td>
<td>70.08</td>
<td>0.580</td>
<td>86.64</td>
<td>0.798</td>
</tr>
<tr>
<td>digitsv27</td>
<td>10</td>
<td>88.20</td>
<td>0.723</td>
<td>84.27</td>
<td>0.734</td>
<td>80.62</td>
<td>0.687</td>
</tr>
<tr>
<td>ionosphere</td>
<td>10</td>
<td>69.52</td>
<td>0.575</td>
<td>64.10</td>
<td>0.539</td>
<td>70.94</td>
<td>0.587</td>
</tr>
<tr>
<td>letterAVB</td>
<td>10</td>
<td>92.80</td>
<td>0.866</td>
<td>94.21</td>
<td>0.891</td>
<td>90.29</td>
<td>0.825</td>
</tr>
<tr>
<td>satellite</td>
<td>16</td>
<td>95.35</td>
<td>0.911</td>
<td>97.45</td>
<td>0.950</td>
<td>95.53</td>
<td>0.915</td>
</tr>
<tr>
<td>digits0689</td>
<td>20</td>
<td>54.84</td>
<td>0.735</td>
<td>93.41</td>
<td>0.600</td>
<td>75.32</td>
<td>0.863</td>
</tr>
<tr>
<td>digits1279</td>
<td>20</td>
<td>74.65</td>
<td>0.811</td>
<td>89.97</td>
<td>0.583</td>
<td>79.53</td>
<td>0.834</td>
</tr>
<tr>
<td>letterABCD</td>
<td>10</td>
<td>66.09</td>
<td>0.773</td>
<td>62.08</td>
<td>0.731</td>
<td>65.67</td>
<td>0.777</td>
</tr>
<tr>
<td>mnist01234</td>
<td>50</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>71.32</td>
<td>0.811</td>
</tr>
<tr>
<td>Acc</td>
<td>89.98</td>
<td>0.901</td>
<td>98.48</td>
<td>0.971</td>
<td>98.75</td>
<td>0.975</td>
<td></td>
</tr>
<tr>
<td>RI</td>
<td>90.85</td>
<td>0.919</td>
<td>90.85</td>
<td>0.919</td>
<td>90.85</td>
<td>0.919</td>
<td></td>
</tr>
</tbody>
</table>

Table 2. Clustering accuracy (%) and Rand Index comparisons on the various data sets. For each method, the number on the left denotes the clustering accuracy, and the number on the right stands for the Rand Index. The symbol '-' means that the corresponding algorithm cannot handle the data set in reasonable time.

<table>
<thead>
<tr>
<th>Data</th>
<th>from whole set</th>
<th>from data subset</th>
<th>subset size</th>
<th>Acc</th>
<th>RI</th>
</tr>
</thead>
<tbody>
<tr>
<td>letterAVB</td>
<td>96.33</td>
<td>0.929</td>
<td>500</td>
<td>95.60</td>
<td>0.912</td>
</tr>
<tr>
<td>satellite</td>
<td>98.75</td>
<td>0.975</td>
<td>500</td>
<td>98.57</td>
<td>0.972</td>
</tr>
<tr>
<td>letterABCD</td>
<td>85.53</td>
<td>0.867</td>
<td>500</td>
<td>83.98</td>
<td>0.852</td>
</tr>
<tr>
<td>mnist01234</td>
<td>90.85</td>
<td>0.919</td>
<td>1000</td>
<td>89.11</td>
<td>0.902</td>
</tr>
</tbody>
</table>

Table 3. Generalization ability on unseen samples when the M3FS model is learned only from a data subset.

Laplacian regularizer in the objective to enforce smoothness on the clustering function. Moreover, we also extend the M3FS algorithm to the multi-class setting. Finally, experimental results on both toy and real-world data sets demonstrate the effectiveness of the proposed approach.

Acknowledgements

Supported by NSFC (Grant No. 60835002 and No. 60675009), and Research Grants Council of the Hong Kong Special Administrative Region under grant 614907.

References