# Adaptive *K*-Nearest Neighbor Classifier Based on Features Extracted by Nonparametric Model

Jinn-Min Yang[1], Pao-Ta Yu[2]

*Department of Computer Science and Information Engineering, National Chung Cheng University*
*168 University Rd., Min-Hsiung 621, Chia-yi, Taiwan, R.O.C.*

[1]ygm@ms3.ntcu.edu.tw
[2]csipty@cs.ccu.edu.tw

***Abstract*—In general there are two main approaches for overcoming the high-dimensional and small sample size (SSS) problem. One is to apply feature extraction or selection to reduce the dimensionality, and then applying the reduced-dimensionality data set to classifier. The other is to modify the classifier design to be suitable for SSS problem. This study integrates the two approaches into a new *K*-nearest neighbour (*K*NN) classifier, namely adaptive *K*NN (A*K*NN). One remotely sensed hyperspectral benchmark image data set is included for investigating the effectiveness of A*K*NN. Experimental results demonstrate that the proposed A*K*NN can perform better than *K*NN and support vector machine (SVM) classifier.**

***Keywords*— *K*-nearest neighbor classifier, dimension reduction, feature extraction, small sample size problem, curse of dimensionality.**

## 1. INTRODUCTION

The so-called small sample size (SSS) problem [1], [2], states that the number of available training samples is much smaller than the dimensionality of the sample space, and has been an important issue for high-dimensional data classification. The Hughes phenomenon [3], [4] (or the curse of dimensionality) clearly describes the increase of the number of dimension potentially increases the class separability and the classification accuracy, but the accuracy will eventually decline when the ratio of the number of the training pixels and the dimensionality cannot be maintained at or above some minimum value to achieve statistical confidence [5]. Learning algorithms suffer from the SSS problem easily, and yield unsatisfactory classification results. The Hughes phenomenon shows two directions for mitigating the SSS problem. One is to reduce the dimensionality by feature extraction or feature selection techniques [6]-[14], and the other is to increase the number of training samples such as semi-supervised techniques [15], [16]. We focus on the application of feature extraction model in this study.

The main purpose of feature extraction or feature selection is to mitigate the Hughes phenomenon. The feature selection method aims to select a suitable subset of the original features. The most important issue relative to feature selection is to find an efficient search strategy for obtaining such a subset for classification. Most of the existing feature selection methods are generally suboptimal due to the number of all possible combinations is prohibitive, particularly for high-dimensional data classification. The search strategies to avoid the exhaustive search are needed, and the selection of the optimal subset is therefore not guaranteed. Feature extraction uses all the features to construct a transformation that maps the original data to a low-dimensional subspace. The main advantage of feature extraction above feature selection is that no information of the original features needs to be wasted. Furthermore, feature extraction is easier than feature selection in some situations [17].

Linear discriminant analysis (LDA) [1] has been played an important role for data classification. It is one of the most well-known dimension reduction methods and has been successfully applied to many classification problems. The purpose of LDA is to find a linear transformation that can be used to project data from a high-dimensional space into a low-dimensional subspace. Basically, LDA has three inherent deficiencies in dealing with classification problems. First, LDA is only well-suited for normally distributed data [1]. If the distributions are significantly non-normal, the use of LDA cannot be expected to accurately indicate which features should be extracted to preserve complex structures needed for classification.

Second, since the rank of between-class scatter matrix is the number of classes ($L$) minus one [1], the number of features can be extracted at most remains the same. Third, the singularity problem arises when dealing with high-dimensional and SSS data. Generally, there are three categories for solving the singularity of within-class scatter matrix [18]. In recent years, many approaches have been proposed to deal with the singularity problem for different applications, including regularized LDA (RLDA) [11], LDA/GSVD [12], LDA/QR [13], nonparametric weighted feature extraction (NWFE) [9], and nonparametric linear discriminant analysis (NLDA) [8]. Regularization and eigen-decomposition are the most often used techniques for alleviating the SSS problem. However, the first two problems still exist. Nonparametric linear discriminant analysis such as nonparametric discriminant analysis (NDA) [14], NWFE and NLDA provides a solution for circumventing both of the previously mentioned problems. In NWFE, a regularization technique is employed to solve the singularity problem, and all problems of LDA are then resolved. Additionally, nonparametric feature extraction is generally of full rank which provides the ability to specify the number of extracted features desired and works well even for non-normally distributed data [8], [9].

Our previous work demonstrated that 1NN and SVM with NLDA features can reach satisfactory performance on high-dimensional data set classification. In this paper, a novel *K*-nearest neighbour (*K*NN) classifier is proposed, namely adaptive *K*NN (A*K*NN), which is constructed based on the NLDA features and the idea of fuzzy *K*NN (F*K*NN) algorithm [19]. In other words, the feature extraction algorithm can not only for reducing dimension but also for building classifier. The effectiveness of the proposed A*K*NN is evaluated by a benchmark hyperspectral data set with different training sample sizes, including the ill-posed and poorly posed classification problems [20].

The rest of the paper is organized as follows. In Section 2, some related work is reviewed. Then the details of the proposed A*K*NN are described in Section 3, followed by experimental designs and results in Section 4. Finally, conclusions are drawn in Section 5.

## 2. RELATED WORK

In this section, some work related to ours is reviewed. For convenience, some important notations employed in the study are presented in Table 1.

**TABLE 1.**
**IMPORTANT NOTATIONS EMPLOYED IN THE PAPER**

| Notation | Description | Notation | Description |
|---|---|---|---|
| $X$ | data matrix | $N$ | total training samples |
| $X_i$ | data matrix of the $i$th class | $N_i$ | number of training samples in the $i$th class |
| $L$ | number of classes | $x_\ell^{(i)}$ | the $\ell$th sample in the $i$th class |
| $P_i$ | prior probability of the $i$th class | $p$ | dimensionality of the reduced subspace |
| $d$ | dimensionality of the original space | $M_j(x_\ell^{(i)})$ | local mean of $x_\ell^{(i)}$ in the $j$th class |
| $A$ | transformation matrix | $\mu$ | regularization parameter |
| $K_m$ | number of nearest neighbors included for estimating the individual metric of each training sample in A*K*NN | | |

### 2.1. Nonparametric Linear Discriminant Analysis (NLDA)

The goal of linear feature extraction is to find a transformation matrix $A$ which maximizes between-class ($S_b$) and minimizes the within-class ($S_w$) scatter matrices in the reduced dimensional space [1]. The common optimization criterion for finding $A$ is

$$A = \underset{A}{\operatorname{argmax}} \operatorname{tr}((A^T S_w A)^{-1} A^T S_b A). \quad (1)$$

The maximization of (1) is equivalent to solving the generalized eigenvalue decomposition problem

$$S_b v_h = \lambda_h S_w v_h, \quad h = 1, \ldots, p. \quad (2)$$

where $p$ denotes the dimensionality of the reduced subspace, $(\lambda_h, v_h)$ represent the eigen-pair of $S_w^{-1} S_b$, and $\lambda_1 \geq \lambda_2 \geq \cdots \geq \lambda_p$. Thus, the transformation matrix $A = [v_1, \ldots, v_p]$ can be obtained.

The within-class scatter matrix of NLDA (denoted as $S_w^G$) is defined as

$$S_w^G = \sum_{i=1}^{L} P_i \sum_{\ell=1}^{N_i} (x_\ell^{(i)} - M_i(x_\ell^{(i)}))(x_\ell^{(i)} - M_i(x_\ell^{(i)}))^T, \quad (3)$$

where $P_i$ and $M_i(x_\ell^{(i)})$ are the prior probability and local mean with respect to $x_\ell^{(i)}$ in class $i$, respectively. The local mean $M_i(x_\ell^{(i)})$ is defined as

$$M_i(x_\ell^{(i)}) = \frac{1}{k}\sum_{s=1}^{k} x_{sNN}^{(i)}, \tag{4}$$

denotes the sample mean of the *K*NNs with respect to $x_\ell^{(i)}$.

The between-class scatter matrix of NLDA ($S_b^G$) is defined as

$$S_b^G = \sum_{i=1}^{L} P_i \sum_{\substack{j=1 \\ j\neq i}}^{L} \sum_{\ell=1}^{N_i} (x_\ell^{(i)} - M_j(x_\ell^{(i)}))(x_\ell^{(i)} - M_j(x_\ell^{(i)}))^T, \tag{5}$$

where $M_j(x_\ell^{(i)})$ is the local mean with respect to $x_\ell^{(i)}$ in class $j$.

The idea to construct NLDA is twofold: First, we find that the local mean $M_i(x_\ell^{(i)})$ can be regarded as a leave-($N_i - k$)-out mean vector. Intuitively, $M_i(x_\ell^{(i)})$ can approximate to class mean $m_i$ as the value of $k$ is close to $N_i$. The estimators of scatter matrices will be more general and flexible. Second, to work well for non-normally distributed data, the within-class and between-class scatter matrices should be nonparametric simultaneously. The geometric depiction of the relationships of the within-class and between-class scatter matrices for the proposed NLDA is demonstrated in Fig. 1. The orange and green dash lines show the relationships between local means and class means in within-class and between-class, respectively.

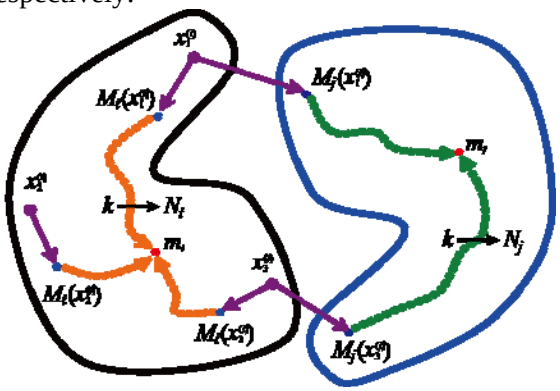


Fig. 1. Geometric depiction on the relationships between the local and class means.

For extracting informative features, the criterion $J = tr(S_w^{-1}S_b)$ requires the within-class scatter matrix $S_w$ to be nonsingular [1], [21]. However, when the size of training samples is small, $S_w$ is often singular or nearly singular. For preventing the singularity of $S_w$, the regularization is one of the prominent techniques [8], [9], [10]. For NLDA, the regularization form is adopted as

$$S_w^{G_R} = (1-\mu)S_w^G + \mu \mathrm{diag}(S_w^G), \tag{6}$$

where $\alpha$ is the regularization parameter. The grid-search and cross validation (CV) methods are adopted to search the best value of $\mu$ in this study.

Simultaneous diagonalization of two matrices is a very powerful tool in pattern recognition [1]. In fact, the transformation matrix $A$ consists of eigenvectors of $(S_w^{G_R})^{-1}S_b^G$ can diagonalize $S_w^{G_R}$ and $S_b^G$ simultaneously, which has been proven in [1, p.32]. Nevertheless, when the singularity problem of $S_w^G$ has resolved by utilizing $S_w^{G_R}$, there is another essential issue about the eigenvectors has to be taken care. That is, the matrix $(S_w^{G_R})^{-1}S_b^G$ may be not symmetric in general, and subsequently the eigenvectors $v_i$'s are not mutually orthogonal. Thus, to make the $v_i$'s orthonormal with respect to $S_w^{G_R}$ to satisfy $A^T S_w^{G_R} A = I$, the scale of $v_i$ must be adjusted by

$$v_i = \frac{v_i}{\sqrt{v_i^T S_w^{G_R} v_i}} \tag{7}$$

such that

$$\frac{v_i^T}{\sqrt{v_i^T S_w^{G_R} v_i}} S_w^{G_R} \frac{v_i}{\sqrt{v_i^T S_w^{G_R} v_i}} = 1. \tag{8}$$

The details of the NLDA is shown in Algorithm 1.

**Algorithm 1:** NLDA

**Input**: the data matrix $X \in R^{d\times N}$, where $d$ is the dimensionality of original space and $N$ is the number of training samples.

**Output:** the projection data matrix $Y = A^T X \in R^{p\times N}$, where $A \in R^{d\times p}$ and $p$ is the dimensionality of reduced subspace.

**Process:**

Step 1. Select a value of $k$ for estimating the local mean $M_j(x_\ell^{(i)})$ with respect to each training sample $x_\ell^{(i)}$ in $X$.

Step 2. Compute the within-class and between-class scatter matrices in (3) and (5), respectively.

Step 3. Calculate the regularized within-class scatter matrix $S_w^{G_R}$ in (6).

Step 4. Select the $p$ eigenvectors of $(S_w^{G_R})^{-1}S_b^G$, which correspond to the $p$ largest eigenvalues.

Step 5. Adjust each eigenvector $v_i$ by (7), $i = 1, \ldots, p$, and $A = [v_1, \ldots, v_p] \in R^{d\times p}$.

Step 6. Calculate the transformed data $Y = A^T X$.

## 2.2. Support Vector Machine (SVM)

The support vector machine (SVM) [22] has considered a must try since it offers one of the most robust and accurate methods among all well-known algorithms [23]. As we know, SVM has a solid theoretical foundation and requires only a dozen samples for training. SVM attempts to separate samples between classes by maximizing the margins in the kernel space where samples are mapped. Fundamentally, the SVM classifier is designed for two-class problems. It can be extended for multiclass problems by designing a number of two-class SVMs. One against one (OAO) and One against all (OAA) are two different approaches.

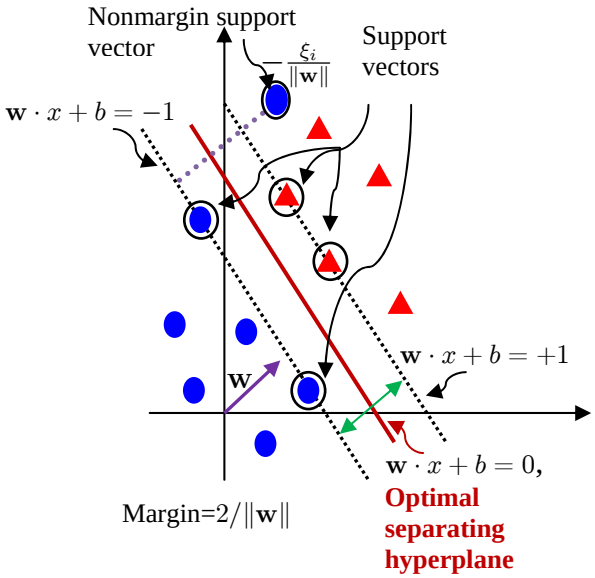


Fig. 2 Optimal separating hyperplane in SVM for a linearly separable case. Red and blue samples refer to the classes "+1" and "-1," respectively. Support vectors are indicated by an extra circle.

Let $D=\{(x_1,y_1),\ldots,(x_N,y_N)\}$ be the training data set which contains $N$ data points, where $y_i\in\{+1,-1\}$ denotes the class label for the data point $x_i$. The problem of finding the weight vector $\mathbf{w}$ can be formulated as the minimization of the following function:

$$L(\mathbf{w})=\frac{1}{2}\|\mathbf{w}\|^2+C\sum_i\xi_i \qquad (9)$$

subject to $y_i[\mathbf{w}\cdot\phi(x_i)+b]\geq 1-\xi_i,\xi_i\geq 0$. Here, the $\xi_i$ is the so-called slack variable and $b$ is the bias and the function $\phi(x)$ maps the input vector to the feature vector. The dual formulation is given by maximizing

$$Q(\lambda)=\sum_{i=1}^{N}\lambda_i-\frac{1}{2}\sum_{i=1}^{N}\sum_{j=1}^{N}y_iy_j\lambda_i\lambda_j\kappa(x_i,x_j), \qquad (10)$$

subject to $\sum_{i=1}^{N}y_i\lambda_i=0$ and $0\leq\lambda_i\leq C$. The parameter $C$, called as regularization parameter, controls the trade off between complexity of the SVM and the misclassification rate. $\kappa(x_i,x_j)=<\phi(x_i),\phi(x_j)>$ is the kernel function. Only a small fraction of the $\lambda_i$'s are nonzero. The corresponding pairs of $x_i$'s are known as support vectors, and they fully define the decision function. Geometrically, the support vectors are the points lying near the separating hyperplane.

There are two main fascinating properties of SVM in practical applications. First, the linear patterns can be represented efficiently via kernel trick [24] without computing their coordinates explicitly; in other words, the algorithms can be implemented in terms of pairwise inner products in feature space and the inner products can be calculated directly from the original data by employing a kernel function. Second, a linear relationship can be found in the feature space, which is equivalent to seeking the nonlinear relationship in the original space.

## 2.3. *K*-Nearest Neighbor Classifier

The *K*-nearest neighbors (*K*NN) classifier is one of the simplest and rather trivial classifiers. *K*NN classifier finds a group of *K* samples in the training set that are close to the test sample, and then the test sample is classified by the majority category of *K*-nearest neighbors. In other words, to classify an unlabeled sample, the distance of this sample to the entire training data is computed, its *K*NNs are identified, and the class labels of these *K*NNs are then used to determine the class label of the test sample. The Euclidean distance is the most widely used similarity (or dissimilarity) metric for *K*NN classifier. The *K*NN classification algorithm is stated in Algorithm 2.

**Algorithm 2: *K*NN Classifiier**

**Input**:

The training set $D=\{(x_1,y_1),\ldots,(x_N,y_N)\}$ and a test sample $x^*=(z,l)$.

**Output**: $l=\underset{v}{\text{argmax}}\sum_{(x_i,y_i)\in D}I(v=y_i)$, where $I(\cdot)$ is an indicator function that returns the value 1 if its argument is true and 0 otherwise.

**Process**:

Step 1. Compute the distance between $x^*$ and every training sample $x_i$, $d(x^*,x_i)$.

Step 2. Find the $K$ closet training samples to $x^*$.

## 3. ADAPTIVE *K*-NEAREST NEIGHBOR CLASSIFICATION (A*K*NN)

The aforementioned *K*NN classifier does not include the training phase since there is no information learned from the training samples before classifying a test sample. In F*K*NN, its training phase, i.e., fuzzification scheme, is devoted to collect local information surrounding each sample, and then the information will exert influence in the classification phase. The local information accompanying each training sample is a membership vector, containing degrees that each training sample belongs to other classes. This is quite an attractive idea for supervised classification, and A*K*NN will embed this idea in it.

One of the important advantages of nonparametric feature extraction methods is that it is generally of full-rank. Thus, the number of features that can be extracted is the same as the dimensionality of the original space. In other words, one can construct another feature space by using nonparametric feature extraction model. In the new feature space, the projected data is more separable than in the original space. In other words, the new feature space forms a better distance metric. Based on the idea of the fuzzification scheme of F*K*NN, we develop A*K*NN. In A*K*NN, each training $x$ carries a distance metric consisting of the all features extracted by NLDA using its nearest $K_m$ samples.

Let $R(x, K_m)$ be the set including $K_m$ nearest training samples with respect to any training sample $x$, and $L^* \le L$ is the number of classes in $R(x, K_m)$. Hence, the local between-class ($S_b^A$) and within-class scatter ($S_w^A$) matrices of $x$ are calculated by

$$\mathcal{S}_w^A(x) = \sum_{i=1}^{L^*} P_i \sum_{x_\ell^{(i)} \in R(x,K_m)} \left(x_\ell^{(i)} - M_i(x_\ell^{(i)})\right)\left(x_\ell^{(i)} - M_i(x_\ell^{(i)})\right)^T, \quad (11)$$

$$\mathcal{S}_b^A(x) = \sum_{i=1}^{L^*} P_i \sum_{\substack{j=1, \\ j\neq i}}^{L^*} \sum_{x_\ell^{(i)} \in R(x,K_m)} \left(x_\ell^{(i)} - M_j(x_\ell^{(i)})\right)\left(x_\ell^{(i)} - M_j(x_\ell^{(i)})\right)^T, \quad (12)$$

where $n_i$ denotes the training sample size of class $i$ in $R(x, K_m)$, and $n_1 + n_2 + \cdots + n_{L^*} = K_m$. Notably, the estimation of the local metric will seriously encounter the SSS problem. If $N_i$ is already small, then the number of the training samples per class in $R(x, K_m)$, denoted as $\hat{N}_i$, is definitely smaller than $N_i$. For example, if $K_m = 30$ and $L^* = 3$, then $\hat{N}_i = 10$ on average. Thus, a more severe condition for extracting features will arise. Importantly, the proposed NLDA has proved to be useful for alleviating this problem.

Let $(\lambda_h, v_h)$ denote the eigen-pair of $(S_w^A(x))^{-1} S_b^A(x)$, $h = 1, 2, \ldots, d$ and $\Lambda = \sum_{i=1}^{d} \lambda_i$. Define a new metric

$$\Sigma_x^A = \frac{\lambda_1}{\Lambda} v_1 v_1^T + \frac{\lambda_2}{\Lambda} v_2 v_2^T + \cdots + \frac{\lambda_p}{\Lambda} v_d v_d^T. \quad (13)$$

Note that

$$\Sigma_x^A v_h = \frac{\lambda_h}{\Lambda} v_h, \forall h = 1, 2, \ldots, d. \quad (14)$$

Thus, $(\frac{\lambda_h}{\Lambda}, v_h)$ is a eigen-pair of $\Sigma_x^A$. In the training phase of A*K*NN, each training sample is with a distance metric (13).

When classifying a test sample $x^*$, the metric $\Sigma_*^A$ is determined by the summation of the weighted metric of its $s$-nearest training neighbors $x_1, \ldots, x_s$ according to

$$\Sigma_*^A = \frac{\sum_{\ell=1}^{s} w_\ell \Sigma_\ell^A}{s}, \quad (15)$$

where $w_\ell = d(x_\ell, x^*)^{-1} / \sum_{\ell=1}^{s} d(x_\ell, x^*)^{-1}$.

Evidently, we weigh $\Sigma_*^A$ by the inverse of the Euclidean distance from $x_\ell$ to $x^*$, where $s$ denotes the number of nearest neighbors of $x^*$. The estimator of $\Sigma_*^A$ in (15) provides a scenario to handle the measurement uncertainty of the distance metric of $x^*$. By applying $\Sigma_*^A$, the new distance between $x$ and a test sample $x^*$ is calculated by

$$\hat{d}(x, x^*) = (x - x^*)^T \Sigma_*^A (x - x^*). \quad (16)$$

After recomputing the distance according to (16), $x^*$ is classified through means of the majority category of its new *K*-nearest neighbors.

The A*K*NN classification algorithm is summarized in Algorithm 3, and its idea is illustrated in Fig. 3.

**Algorithm 3: A*K*NN Classifier**

**Input**:

The training set $D = \{(x_1, y_1), \ldots, (x_N, y_N)\}$ and a test sample $x^*$.

**Output**: $l = \underset{v}{\operatorname{argmax}} \sum_{(x_i, y_i) \in D} I(v = y_i)$, where $I(\cdot)$ is an indicator function that returns the value 1 if its argument is true and 0 otherwise.

**Process**:

**A. Training phase**

Step 1. Given the value $K_m$ and find the $K_m$-nearest neighbors for each training sample $x_\ell$ according to Euclidean distance, $\ell = 1, 2, \ldots, N$.

Step 2. Calculate the local within-class

($S_w^A(x_\ell)$) and between-class ($S_b^A(x_\ell)$) scatter matrices of $x_\ell$ by using the samples in $R(x_\ell, K_m)$.

Step 3. Estimate the metric $\Sigma_\ell^A$ of $x_\ell$, $\ell = 1, 2, \ldots, N$.

**B. Classification phase**

Step 1. Estimate the metric $\Sigma_*^A$ for a test sample $x^*$ according to (15).

Step 2. Compute the new distance between $x_\ell$ and $x^*$ by $(x_\ell - x^*)^T \Sigma_*^A (x_\ell - x^*)$, $\ell = 1, 2, \ldots, N$.

Step 3. Classify $x^*$ by the majority category of the new *K*-nearest training neighbors.

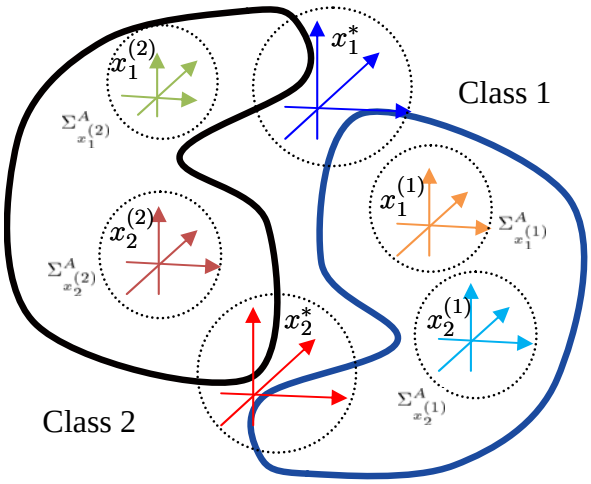


Fig. 3 Illustration of A*K*NN.

## 4. Experimental Design and Results

### 4.1. Data Set

For evaluating the performance of the proposed A*K*NN, one real hyperspectral image dataset, Indian Pines scene (IPS), is included. The IPS image was gathered by the AVIRIS instrument in 1992, mounted from an aircraft flown at 65000 ft. altitude and operated by the NASA/Jet Propulsion Laboratory, with the size of 145×145 pixels has 220 spectral bands measuring approximately 20m across on the ground. The data set represents a very challenging land-cover classification scenario, in which the primary crops of the area (mainly corn and soybeans) were very early in their growth cycle, with only about 5% canopy cover. Discriminating among the major crops under these circumstances can be very difficult. The IPS consists of 16 ground-truth classes, ranging from 20 to 2468 pixels in size. Since the size of samples in some classes are too small to retain enough disjoint samples for training and testing, only nine classes, Corn-min, Corn-notill, Grass/Pasture, Grass/Tree, Hay-windrowed, Soybeans-min, Soybeans-clean, Soybeans-notill, and Woods, were selected for the experiments. The data is available online from http://cobweb.ecn.purdue.edu/~biehl/MultiSpec/. All the Indian Pines data samples are divided into 4757 training samples and 4588 test samples, as listed in Table 2. Experiments with four different sets containing 2%, 5%, 10% and 25% of the training samples are carried out.

**TABLE 2.**
**NUMBER OF TRAINING AND TEST SAMPLES USED IN THE INDIAN PINES SCENE.**

| No | Name | #Train | #Test |
|---|---|---|---|
| 1 | Corn-notill | 742 | 692 |
| 2 | Corn-min | 442 | 392 |
| 3 | Grass/Pasture | 260 | 237 |
| 4 | Grass/Trees | 389 | 358 |
| 5 | Hay-windrowed | 236 | 253 |
| 6 | Soybeans-notill | 487 | 481 |
| 7 | Soybeans-min | 1245 | 1223 |
| 8 | Soybeans-clean | 305 | 309 |
| 9 | Woods | 651 | 643 |
| | | 4757 | 4588 |

### 4.2. Experimental Design

The 2% and 5% cases are the so-called ill-posed and poorly posed classification problems [20], respectively. They are challenging cases in the field of pattern recognition. Two other classifiers, the 1-nearest neighbour (1NN) and soft-margin SVM with RBF kernel function (SVM-RBF) classifiers, are used for comparison in this study, which are implemented in PRTools [25] and LIBSVM [26], respectively. The 1NN and SVM are considered two of the robust classifiers in the pattern recognition field. For the soft-margin SVM classifier, there is a parameter $C$ to control the trade-off between the margin and the size of the slack variables, and a parameter $\sigma$ for the RBF kernel function. We use the five-fold cross validation to find the best $C$ and $\sigma$ within the given set $\{10^{-5}, \ldots, 10^5\}$. The values of $k$ for estimating the local mean in NLDA is set to 3.

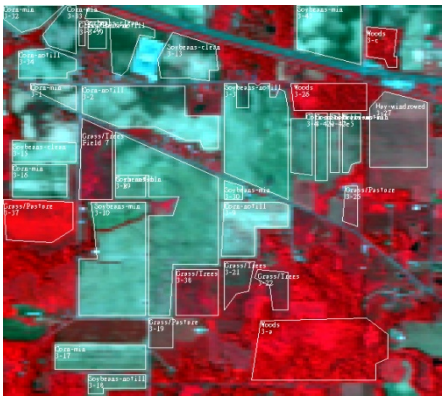

Fig. 4 The test image of the Indian Pines data set. Bands 50, 27 and 17 of 220 bands were used for this image space presentation.

### 4.3. Experimental Results

Table 3 lists the classification results on the IPS data set. A*K*NN outperforms *K*NN and SVM in all cases. A*K*NN significantly outperforms *K*NN, which shows that using fuzzy local metric is useful than the Euclidean metric. There is about 10% difference between A*K*NN and *K*NN in 10% and 25% cases. Also, there is more than 3% difference between A*K*NN and SVM in 5%, 10% and 25% cases.

**TABLE 3**
**THE CLASSIFICATION ACCURACIES ( IN %) OF USING 1NN, SVM-RBF AND A*K*NN CLASSIFIERS.**

| Classifier | 2% | 5% | 10% | 25% |
|---|---|---|---|---|
| *K*NN | 55.9 | 69.1 | 71.9 | 77.5 |
| SVM | 61.7 | 73.6 | 78 | 84.4 |
| A*K*NN | **62.5** | **77.3** | **81** | **87.9** |

Fig. 5 is the ground truth of Fig. 4. Some IPS thematic maps classified by the three classifiers are demonstrated in Figs. 6 to 11. The results in the 10% case are summarized as follows:

1) A*K*NN achieves the best visual effect, particularly in the red squared area, including "Corn-notill", "Soybeans-clean", and "Soybeans-min" parts.
2) The blue circled part shows that the misclassification difference between SVM and A*K*NN. In this part, SVM tends to misclassify "Soybeans-notill" to "Soybeans-min", but A*K*NN tends to misclassify "Soybeans-notill" to "Corn-notill". The "Soybeans-notill" part at the bottom of the figure (see Fig. 5) is almost completely misclassified to "Soybeans-min" for SVM.

The results in the 25% case are similar to the 10% case. A*K*NN still achieves the best visual effect, and has better classification in almost every part.


Fig. 5 The ground truth of Fig.4

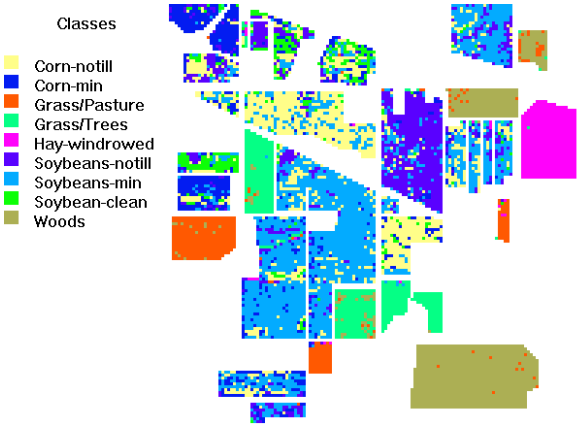


Fig. 6 Thematic maps resulting from the classification of the area of Fig. 5 in the 10% case by 1NN classifier.

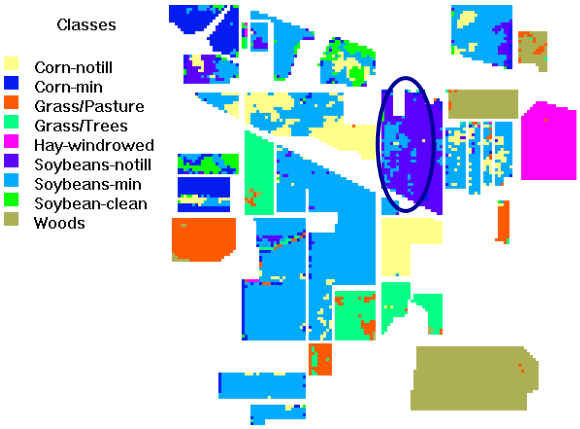


Fig. 7 Thematic maps resulting from the classification of the area of Fig. 5 in the 10% case by SVM classifier.

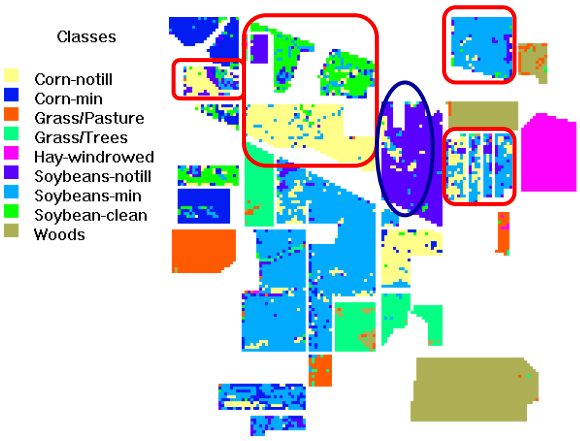


Fig. 8 Thematic maps resulting from the classification of the area of Fig. 5 in the 10% case by A*K*NN classifier.

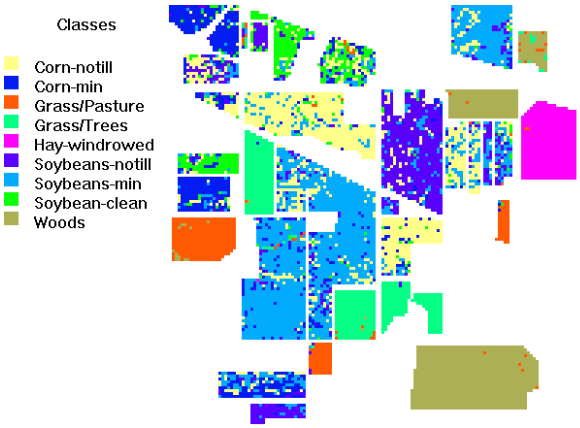


Fig. 9 Thematic maps resulting from the classification of the area of Fig. 5 in the 25% case by 1NN classifier.

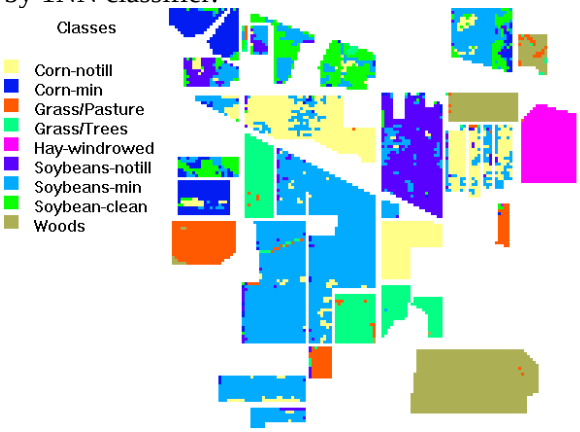


Fig. 10 Thematic maps resulting from the classification of the area of Fig. 5 in the 25% case by SVM classifier.

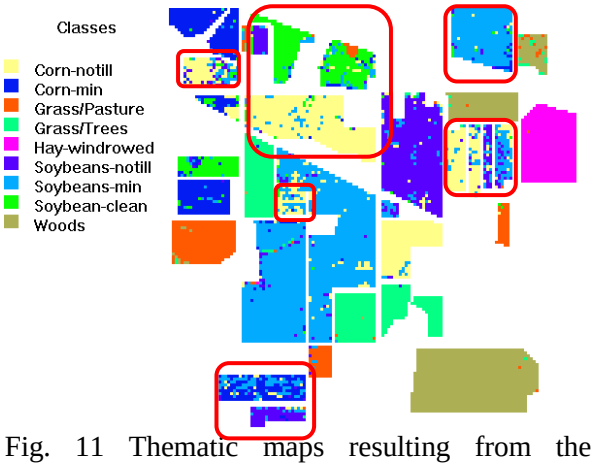


Fig. 11 Thematic maps resulting from the classification of the area of Fig. 5 in the 25% case by A*K*NN classifier.

## 5. Conclusions

The objective of this study is to introduce a new application of nonparametric feature extraction method for building a *K*NN-type classifier. The proposed A*K*NN contains training and classification phases, in which the advantage of nonparametric feature extraction model is taken in the training phase and the concept of uncertainty is embedded in the classification phase. The experimental results demonstrated that it significantly outperformed the classic *K*NN classifier. A*K*NN also obtained satisfactory results as compared with SVM.

## References

[1] K. Fukunaga, Introduction to Statistical Pattern Recognition, second ed., Academic Press, New York, 1990.

[2] S.J. Raudys and A.K. Jain, “Small sample size effects in statistical pattern recognition: recommendations for practitioners,” IEEE Transaction on Pattern Analysis and Machine Intelligence, vol.13 no.3, pp. 252-264, 1991.

[3] G.F. Hughes, “On the mean accuracy of statistical pattern recognizers,” IEEE Transactions on Information Theory, vol. 14, no. 1, pp. 55-63, 1968.

[4] D.A. Landgrebe, Signal Theory Methods in Multispectral Remote Sensing, John Wiley and Sons, Hoboken, Chichester, 2003.

[5] P.K. Varshney and M.K Arora. Advanced image processing techniques for remotely sensed hyperspectral data, Springer, New York, 2004.

[6] S. B. Serpico and L.Bruzzone, “A new search algorithm for feature selection in hyperspectral remote sensing images,” IEEE Transactions on Geoscience and Remote Sensing, vol. 39, no. 7, pp.1360-1367, Jul. 1994.

[7] S. B. Serpico and G. Moser, “Extraction of spectral channels from hyperspectral images for classification purposes,” IEEE Transactions on Geoscience and Remote Sensing, vol. 45, no. 2, pp.484-495, Feb. 2007.

[8] J.M. Yang, P.T. Yu,” A Novel Nonparametric Linear Discriminant Analysis for High-Dimensional Data Classification,” 2009 International Conference on Advanced Information Technologies, Chaoyang University of Technology, Taichung, Taiwan.

[9] B.C. Kuo and D.A. Landgrebe, “Nonparametric weighted feature extraction for classification,” IEEE Transaction on Geoscience and Remote Sensing, vol.42, no.5, pp.1096-1105, 2004.

[10] J.M. Yang, P.T. Yu, B.C. Kuo, T.Y. Hsieh, “A novel fuzzy linear feature extraction for hyperspectral image classification,” in Proc. of IEEE International Conference on Geoscience and Remote Sensing Symposium, 2006, pp. 3895 - 3898.

[11] Y. Guo, T. Hastie, and R. Tibshirani, “Regularized linear discriminant analysis and its application in microarrays,” Biostatistics, vol. 8, no. 1, pp. 86–100, 2007.

[12] P. Howland, M. Jeon, and H. Park, “Structure preserving dimension reduction for clustered text data based on the generalized singular value decomposition,” SIAM J. Matrix Anal. Appl., vol. 25, no. 1, pp. 165–179, 2003.

[13] J. Ye and Q. Li, “LDA/QR: an efficient and effective dimension reduction algorithm and its theoretical foundation,” Pattern Recognition, vol.37, no.4, pp.851-854, 2004.

[14] K. Fukunaga and J.M. Mantock, “Nonparametric discriminant analysis,” IEEE Transaction on Pattern Analysis and Machine Intelligence, vol.5, pp. 671-678, 1983.

[15] L. Bruzzone, C. Mingmin, M. Marconcini, “A Novel Transductive SVM for Semisupervised Classification of Remote-Sensing Images”, IEEE Trans. Geosci. Remote Sens., vol. 44, no. 11, pp. 3363-3373, 2006.

[16] L. Bruzzone and C. Persello, “A novel context-sensitive semisupervised SVM classifier robust to mislabeled training samples,” IEEE Trans. Geosci. Remote Sens., vol. 47, no. 7, pp. 2142-2154, 2009.

[17] F. van der Heiden, R.P.W. Duin, D. de Ridder, and D.M.J. Tax, Classification, Parameter Estimation and State Estimation: An Engineering Approach Using MATLAB, Chichester: John Wiley & Sons, 2004.

[18] A. R. Webb, Statistical Pattern Recognition, second ed., John Wiley and Sons, Hoboken, Chichester, 2002.

[19] J.M. Keller, M.R. Gray, J.A. Givens, A fuzzy k-nearest neighbor algorithm, IEEE Transaction on Systems, Man, and Cybernetics 15 (4) (1985) 580-58.

[20] A. Baraldi, L. Bruzzone, and P. Blonda, “Quality assessment of classification and cluster maps without ground truth knowledge,” IEEE Transactions on Geoscience and Remote Sensing, vol. 43, no. 4, pp.857-873, April, 2005.

[21] R.O. Duda, P.E. Hart, and D.G. Stork, Pattern Classification, second ed., John Wiley & Sons, New York, 2001.

[22] V.N. Vapnik, Statistical Learning Theory. New York: Wiley, 1998.

[23] Wu, X., Kumar, V., Quinlan, J. R., Ghosh, J., Yang, Q., Motoda, H., McLachlan, G. J., Ng, A,. Liu, B., Yu, P. S., Zhou, Z. H., Steinbach, M., Hand, D. J., & Steingberg, D. Top 10 algorithms in data mining. Knowledge and Information Systems, vol. 14, pp.1-37, 2008.

[24] B. Schölkopf, A.J. Smola. Learning with Kernels, The MIT Press, MA, 2002.

[25] R. P. W. Duin, “PRTools, a Matlab toolbox for pattern recognition,” [Online]. Available: http://www.prtools.org/, 2008.

[26] C. C. Chang and C. J. Lin, “LIBSVM : a library for support vector machines,” 2001. [Online]. Available: http://www.csie.ntu.edu.tw/~cjlin/libsvm.