Manifold learning for the shape-based recognition of historical Arabic documents

Mohamed Cheriet, Reza Farrahi Moghaddam, Ehsan Arabnejad and Guoqiang Zhong

Synchromedia Laboratory for Multimedia Communication in Telepresence,
École de technologie supérieure, Montreal, (QC), Canada H3C 1K3
Tel: +1(514)396-8972
Fax: +1(514)396-8595
rfarrahi@synchromedia.ca, imriss@ieee.org, mohamed.cheriet@etsmtl.ca, earabnejad@synchromedia.ca, guoqiang.zhong@synchromedia.ca

Accepted version

Published in: Handbook of Statistics, Vol. 31, ISSN: 0169-7161
DOI: http://dx.doi.org/10.1016/B978-0-444-53859-8.00019-9
URL: http://www.sciencedirect.com/science/article/pii/B9780444538598000199

Abstract

In this work, a recognition approach applicable at the letter block (subword) level for Arabic manuscripts is introduced. The approach starts with the binary images of the letter block to build their input representation, which makes it highly objective and independent of the designer. Then, using two different manifold learning techniques, the representations are reduced and learned. In order to decrease the computational complexity, PCA is applied to the input representations before manifold learning is applied. Also, in order to increase the performance and quality of the input representations, a gray stroke map (GSM) is considered in addition to the binary images. The performance of the approach is tested against a database from

Preprint submitted to Elsevier Science 9 November 2013
1 Introduction

The recognition of text on historical manuscripts is of great interest around the world for understanding the huge volumes of digitized manuscripts that have been produced [3, 2, 5]. The availability of live text not only provides the ability to perform online searching in the manuscript, it boosts the research conducted in the field of history and philosophy by an order of magnitude, and this will have a vast social and environmental impact on the societies involved. However, the recognition and extraction of text from historical manuscripts is not an easy task, considering the wide variations in writing styles, including writer-related variations, as well as degradation. In cases where recognition is not possible, other solutions, such as word-spotting, have been employed to create indices of manuscripts [11, 33, 4].

The situation is a great deal worse for manuscripts in Arabic [24]. This is partly because Arabic script is cursive by nature, which makes it difficult for many of the recognition techniques developed for Latin script to work with it. Also, there are many different writing styles for Arabic script, such as Naskh, Nastaliq, etc., which are not only different in terms of character strokes, they follow completely different rules of calligraphy [28, 29]. For example, the baselines in Nastaliq Arabic, which is a Persian style, are not horizontal lines; moreover, they shift up and down along the text line [42]. In many cases, there are not enough pixels associated with a letter in a letter block for it to be
learned. Also, the imprecise placement of diacritics and dots, especially dots, is a common issue in Arabic styles. Some samples of Nastaliq script are provided in Figure 1 which show aforementioned difficulties, including not horizontal lines and not sequential appearance of letters. The images are courtesy of Nastaliq Script Exhibition\(^1\), Kakayi: The Art Of Oriental Calligraphy\(^2\), and Hamid Akbari Gallery\(^3\).

Recognition is a highly multi-class classification problem at the word level (meaningful level). In the Arabic language, there are more than 280,000 unique words [1]. By moving to the lower level of the letter blocks (subwords or connected components), the number of unique classes drops to 66,000, which reduces the complexity of the classification. It is worth noting that recognition systems usually work at a much lower (character) level in the case of Latin script, which drastically reduces the number of classes to 26. However, other approaches should be considered because in many old manuscripts segmentation of subword images into individual characters, graphemes, or even vertical lines is very difficult because of not sequential nature of their script and also lack of presence of a unique baseline (see Figure 1, for example). As the classification at a scale of 60000 classes is still an unsolvable problem, other approaches could be pursued in order to reduce the number of classes at the letter block level, for example, the binary descriptors approach [14, 15]. In the binary descriptors approach, which we will use in this work, the class definition is kept at the character level, but the samples (observations) are moved to the letter block level [14, 15]. In other words, each object (character) is separately learned, in the form of a binary descriptor, on the samples (the

\(^1\) http://calligraphy.blogfa.com/8505.aspx
\(^2\) http://kakayicalligraphy.webs.com/shekastastyle.htm
\(^3\) http://payameghalam.persianblog.ir/post/1139
letter-blocks), and then all the binary descriptors are combined at the end to retrieve the letters of a letter-block. Usually, a skeleton-based or curve-based representation of the letter-blocks is used in order to reduce their complexity. It has been observed that a few hundreds of binary descriptors are sufficient to learn all possible subwords [15]. These descriptors include the position of a few characters at the beginning of letter blocks in order to reconstruct the whole
letter sequence of the letter block. In other words, in the binary descriptors approach, instead of trying to segment the subword image into character/letter sub images, a set of binary descriptors is defined, which, if learned, could be used to reconstruct the associated string of subwords [14, 15]. Some examples of binary descriptors are: i) the presence of a specific letter in the subword (regardless of its position); ii) the presence of a specific letter as the first letter of the subword; iii) the presence of a specific letter as the second letter of the subword; iv) more than one occurrence of a letter in a subword; and v) the number of letters in a subword.

It is worth noting that subwords processing can also be used for word spotting [10]. As retrieval of the live text is not required in the word spotting applications, no explicit learning is needed, and usually a clustering process can provide spotted instances of a query image based on its shape (subword) level features.

As the learning part of the system is almost independent of the class management part, we ignore the latter, and focus on the learning part, which is a difficult problem in itself. The factors contributing to this learning difficulty are: the variability of the domain (different sizes of letter block images), the wide spectrum of complexity in shapes, the lack of upper and lower profiles because of backward strokes, and non horizontal baselines, among many others.

The most important stage in a learning process is the generation and selection of the appropriate features for representing the objects in an optimal way, i.e. obtaining the maximum amount of information about the problem, while at the same time reducing the noise in the representation. Subword (word)
images are rich in visual information, and this makes feature selection more
difficult. Usually, a fraction of the information available is extracted and used
as features. Skeleton images, in contrast, contain much less noisy information
[26, 38]. However, it is much more difficult to compare them at the image
level, and so it is the high-level features that are usually extracted from them
[46]. For example, in [14], several topological and geometrical descriptors of
variable dimensions have been used to represent a skeleton. Using the contour
of the subword is another means for vectorizing subword images [7, 25]. Other
approaches include using pixel density and a histogram of directions [27]. In
all these approaches, the features are extracted or reduced based on the sub-
jective view of the model designer. Although all these approaches and their
combinations will eventually be improved and converge to produce the opti-
mal set of features in the future, a more objective way to achieve this would be
direct feature reduction. It is, in fact, critical to do this in many applications,
where there is no cue or hint as to they way in which to represent the objects in
raw form, and the number of features is very high. Examples include genome
projects, text categorization, image retrieval, and customer relationship man-
agement [45]. In this work, we apply various approaches to reduce the number
of input features, which are constructed from the normalized binary image of
each subword and one of its generated maps.

Various approaches have been used to reduce the number of features. Both
unsupervised methods such as PCA [30] and ICA [21], and supervised methods
such as LDA [16, 32], have been used for this purpose. All these methods
assume a predefined linear or non-linear model for the data. To arrive at a
more objective approach, we will use manifold learning methods [6, 17] to
rearrange the representations and reduce the number of features. Details of
the manifold learning approaches are presented in section 3.

In this work, a shape-based recognition of Arabic letter blocks (subwords) is investigated. Raw pixel-level features from the binary images and the gray stroke map (GSM) of letter blocks is considered as the initial representation. Then, two different manifold learning approaches, GDA and LLE, are used to reduce the complexity of the representation. The performance of the proposed approach is evaluated against a database of letter blocks from a real manuscript [43].

The paper is organized as follows. In section 2, the problem statement is provided. A brief description of two manifold learning approaches is presented in section 3. A description of the features extracted from the binary images and the GSM are provided in section 4. Experimental results are discussed in section 5. In section 6, our conclusion and some prospects for future work are presented.

2 Problem statement

A ground truth collection of letter block images (a letter block being a series of letters and their associated diacritics and dots) of a writing style of Arabic script is available. Each letter block is associated with its string. The goal is to build a system that can obtain the text of each letter block of that writing style. The system will use the pixel-level features of binary images and a GSMs of the letter blocks. Several manifold learning approaches are used.
3 Manifold learning

Dimensionality reduction is an essential step in high-dimensional data analysis. The dimension reduction algorithms are applied before the classification algorithms, as a data preprocessing step, in order to arrive at a minimal number of features by removing irrelevant, redundant, and noisy information. Large amounts of data, such as the image data, are considered as high-dimensional, and most recognition systems use a linear method that ignores the properties of manifolds. Nonlinear dimensionality reduction methods are commonly used for two purposes in reducing the number of input variables; for extracting the manifold of the most important features and organizing the data for better visualization. Below, two different approaches to manifold learning are discussed.

3.1 Locally-linear embedding (LLE)

Locally-linear embedding (LLE) [34] makes the assumption that each point can be reconstructed locally by its neighbors, and the low-dimensional representation of data can be achieved using reconstruction weights. The process can be summarized in the following three LLE steps:

(1) For each data point, the distances between that point and the others are computed, and its K nearest neighbors are selected.

(2) The reconstruction weights of all points is calculated using K-NN:

\[
\hat{W} = \arg \min_W \epsilon(W) = \arg \min_W \sum_i \|x^i - \sum_j w_{ij} x^j\|^2
\]

\[
s.t. \quad \sum_j w_{ij} = 1
\]
where $\tilde{W}$ is the calculated weight matrix.

(3) The embedding coordinates are computed by minimizing the reconstruction error $\phi$ of the coordinates using $W$:

$$\tilde{Y} = \arg\min_Y \phi(Y) = \arg\min_Y \sum_y ||Y^i - \sum_j w_{ij} Y^j||^2$$

where $\tilde{Y}$ is the embedded representation. A typical example of LLE is shown in Figure 2. Figure 2(a) shows the original data. For the same of presentation, the original data reduced to 3 dimensions using PCA is shown this figure. Figure 2(b) shows the reduction to 2 dimensions using PCA. As it expected, the overlapping between classes is very high. In contrast, Figure 2(c) shows the reduction obtained using the LLE approach. The overlapping is much less, and is expected to be lower when supervised LLE is used.

3.2 Supervised locally linear embedding (SLLE)

LLE is not directly applicable into many pattern recognition problems because it does not consider the known class label information of the input data. The purpose of introducing SLLE (supervised locally linear embedding) is to use special mappings that separate a within-class structure from a between-class structure. One approach to doing this is to add a term to the distance between samples from different classes which only modifies the first step of the original LLE, and leaves the other two steps unchanged. This can be achieved by artificially increasing the pre-calculated Euclidean distance between samples from different classes, but leaving these distances unchanged if the samples
Fig. 2. An example of dimension reduction to two dimensions using LLE. a) The original data. b) Reduction to 2D using PCA. c) Reduction to 2D using LLE.

are from the same class [17]:

\[
D' = D_{Euc} + \alpha \max(D_{ij})(1 - \delta_{ij})
\]  

(3)

where \(\delta_{ij}\) is Kronecker’s delta.

Another method of imposing supervision on LLE is to apply some shrink/expand functions which decrease the in-class distance while increasing the between-
class distance [31]:

\[
D_{ij} = \begin{cases} 
\sqrt{e^{D_{ij}^2/\beta} - \alpha}, & \text{if } i \neq j; \\
\sqrt{1 - e^{-D_{ij}^2/\beta}}, & \text{if } i = j; 
\end{cases}
\]  

(4)

The parameter \(\beta\) is set to the average euclidean distance of all training data, and the values of the parameters \(\alpha\) and \(K\) are set in an optimizing procedure to get a minimum error rate in the training set.

The above algorithms only change the first step of the original LLE algorithm. We will use the new distance (4) in our experiments.

LLE does not provided a straightforward method for the embedding of new data that are not in the training set. This is also the case for the supervised LLE. The supervised LLE only projects the training data, not the test data. In order to apply the learned manifold to a new data point in the feature space, we follow the non-parametric kernel-based approach using the regression weights matrix introduced in [8].

3.3 A measure of manifold quality

The only parameters that should be determined by a human in LLE are the number of nearest neighbors and the embedding dimension. The embedding dimension depends on the number of nearest neighbors. However, because of the nature of the LLE algorithm, we cannot select an embedding dimension larger than the number of nearest neighbors. Selecting \(K\) is a challenge in LLE, because, with a small \(K\), we lose at the global scale, and with a large \(K\) we lose at the local scale. So, we must have some criteria to determine the number of nearest neighbors in an optimal way.
In [23], the residual variance is used to measure the embedding quality:

$$\sigma^2_R = 1 - \rho^2_{D_x D_y}$$  \hspace{1cm} (5)

where $\rho^2$ is the correlation of the distance matrix in the original space and the embedding space. The minimum value for the residual value (sum of the squared errors) corresponds to the best representation.

In [39], a measure of the embedding quality is introduced as follows:

$$C(X,Y) = \frac{1}{2n} \sum_{i=1}^{n} \left\{ \frac{1}{n} \sum_{j=1}^{n} \left[ D(x_i, \eta_j) - D(y_i, \eta_j) \right]^2 + \frac{1}{k_n} \sum_{j=1}^{k_n} \left[ D(x_i, \theta_j) - D(y_i, \gamma_j) \right]^2 \right\}$$  \hspace{1cm} (6)

The first term represents the local properties in the embedding and high-dimensional spaces, and this quantitative measure illustrates how well the distance information is preserved. The second term represents the error that occurs when the points far away in the high-dimensional space are mapped close together in the embedding space, owing to the selection of the incorrect number of nearest neighbors. The procedure for selecting the optimal number of neighbors is as follows:

- Start the LLE algorithm with the initial $K$ and compute the embedding.
- Calculate the embedding quality using the selected criteria.
- Change $K$ until the minimum value of $C$ is achieved that corresponds to the optimal $K$.

3.4 Generalized discriminant Analysis

Generalized discriminant analysis (GDA) [6] is a kernelized variant of lin-
ear discriminant analysis (LDA) [16]. However, unlike LDA, which seeks a \textit{linear} projection that simultaneously minimizes the within-class scatter and maximizes the between-class scatter to separate the classes, GDA pursues a \textit{nonlinear} mapping. Hence, GDA overcomes the limitation of LDA that it can only deliver linear projection of the data.

LDA is a commonly used statistical approach for dimensionality reduction. Suppose we are given $N$ training data, $\mathcal{T} = \{\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_N\}$, where $\mathbf{x}_i \in \mathbb{R}^D$, $i = 1, 2, \ldots, N$. We denote the number of classes as $C$, and the number of samples in class $\Delta_j$ as $n_j$, $j = 1, 2, \ldots, C$. The “between classes scatter matrix”, $\mathbf{S}_B$, and the “within classes scatter matrix”, $\mathbf{S}_W$, are defined as

$$
\mathbf{S}_B = \sum_j n_j (\mu_j - \mu)(\mu_j - \mu)^T, \tag{7}
$$

and

$$
\mathbf{S}_W = \sum_j \sum_{\mathbf{x}_i \in \Delta_j} (\mathbf{x}_i - \mu_j)(\mathbf{x}_i - \mu_j)^T, \tag{8}
$$

where

$$
\mu = \frac{1}{N} \sum_i \mathbf{x}_i, \tag{9}
$$

and

$$
\mu_j = \frac{1}{n_j} \sum_{\mathbf{x}_k \in \Delta_j} \mathbf{x}_k, \tag{10}
$$

are the overall mean of the training data and mean of samples in class $\Delta_j$, respectively. To find the optimum projection matrix, $\mathbf{P}$, LDA maximizes the following objective:

$$
J(\mathbf{P}) = \frac{\text{tr}(\mathbf{P}^T \mathbf{S}_B \mathbf{P})}{\text{tr}(\mathbf{P}^T \mathbf{S}_W \mathbf{P})}, \tag{11}
$$
where \( tr(\cdot) \) is the trace of a square matrix. Since \( J \) is invariant with respect to (w.r.t.) the rescalings of \( P \): \( P \sim \eta P \), we can enforce the denominator to be simply \( tr(P^T S_W P) = 1 \). To the end, we can rewrite the problem of maximizing \( J \) into a constrained optimization problem:

\[
\begin{align*}
\min_P & -tr(P^T S_B P) \\
\text{s.t.} & \quad tr(P^T S_W P) = 1.
\end{align*}
\]  

(12)

Introducing the Lagrangian multiplier, \( \lambda \), the Lagrange function w.r.t. Problem (12) can be written as

\[
\mathcal{L}(\lambda, P) = -tr(P^T S_B P) + \lambda(tr(P^T S_W P) - 1).
\]  

(13)

Minimizing Problem (13) is equivalent to solve a generalized eigenvalue decomposition problem:

\[
S_B P = S_W \Lambda P,
\]  

(14)

where \( \Lambda \) is a diagonal matrix. Since the rank of \( S_B \) is at most \( C - 1 \), the obtained solution that maximizes \( J \) generally includes \( C - 1 \) eigenvectors of Equation (14) corresponding to the nonzero eigenvalues.

Using the so-called kernel trick, GDA adapts LDA to its nonlinear version. The main idea of GDA is to map the input space into a high dimensional (possibly infinite) feature space in which variables are nonlinearly related to the input space. This technique has also been applied to some other algorithms, such as kernel principal component analysis (KPCA) [36] and support vector machines (SVMs) [40, 41]. Let \( \phi(\cdot) \) denote the nonlinear mapping from the data space, \( \mathcal{R}^D \), to the reproducing kernel Hilbert space (RKHS), \( \mathcal{H} \), which corresponds
to a kernel function $k(\cdot, \cdot) = \phi(\cdot)^T \phi(\cdot)$. The “total scatter matrix”, $S_t$, and the “between classes scatter matrix”, $S_b$, are defined as

$$S_t = \sum_i (\phi(x_i) - m)(\phi(x_i) - m)^T,$$

(15)

and

$$S_b = \sum_j n_j(m_j - m)(m_j - m)^T,$$

(16)

where

$$m = \frac{1}{N} \sum_i \phi(x_i),$$

(17)

and

$$m_j = \frac{1}{n_j} \sum_{x_k \in \Delta_j} \phi(x_k),$$

(18)

are the overall mean of the data in the feature space $H$ and class mean of $\Delta_j$, respectively.

To implement LDA in the feature space $H$, GDA optimizes the following trace function w.r.t. $S_t$ and $S_b$:

$$W^* = \arg \max_W \text{tr}(W^T S_t W) - \frac{1}{2} W^T S_b W).$$

(19)

However, since the explicit form of $\phi(\cdot)$ is unknown, Problem (19) cannot be straightforwardly solved via generalized eigenvalue decomposition. Fortunately, according to the representer theorem [22, 35]:

$$W = XP,$$

(20)

where $X = [\phi(x_1), \phi(x_2), \ldots, \phi(x_N)]$ is the data matrix in the feature space, $H$, and $P$ is the coefficient matrix. Thus, Problem (19) can be rewritten as
\[ P^* = \arg \max_P tr((P^T K H_N K P)^{-1}(P^T K H_N G H_N K P)), \]  

(21)

where \( K \) is the kernel matrix, \( H_N = I_N - \frac{1}{N} 1_N 1_N^T \) (\( I_N \) is the \( N \times N \) identity matrix and \( 1_N \) is an \( N \times 1 \) vector of all ones), and \( G \) is a similarity matrix defined as

\[
G(s, t) = \begin{cases} 
\frac{1}{n_j}, & x_s \in \Delta_j \text{ and } x_t \in \Delta_j; \\
0, & \text{otherwise}.
\end{cases}
\]  

(22)

Similar to LDA, the optimal solution \( P^* \) of Problem (21) generally includes \( C - 1 \) eigenvectors of \((K H_N K)^{-1}(K H_N G H_N K)\) corresponding to the nonzero eigenvalues.

4 Feature extraction

As has been mentioned, we use the raw binary image of a letter block as the input representation and features. The color document images of the manuscript are first binarized using the grid-based Sauvola method [13]. After attaching the diacritics and dots to their hosting connected components (CCs), the binary images of letter blocks are extracted. Two examples of a letter block is shown in Figure 3. As will be discussed in the Experimental results section, padding and translation are used to make the binary images the same size. As Arabic script is written from right to left, padding is performed on the left side of the images. Also, the letter block images are centered vertically on their baseline, which is calculated using a modified local averaging on its neighboring CCs.
4.1 Gray stroke map (GSM)

The stroke map (SM) was introduced in [12]. The main concept behind SM is the identification of pixels of interest (POI) that have a stroke structure around them. The SM is a map which assigns a value to each pixel on the input image. In the SM, the finite and fixed width of the pen, which is measured in the form of the average stroke width $w_s$ [13], is the main clue for identifying the stroke pixels. Various implementations of SM have been used from the kernel-based methods [12, 44] to overlap patches [13]. In this work, another implementation of SM, based on the skeleton image and $w_s$, is proposed, in order to reduce the computational time. The details of the implementation are provided in algorithm 1. One of intermediate states in the computation of the SM in this implementation is defined as a new map: the gray stroke map (GSM). This map gives the probability (membership value) of a pixel belonging to the text strokes. An example illustrating this is given in Figures 4 and 5. A rough and over segmented binarization is corrected in the GSM and SM results, thanks to the $w_s$ \textit{a priori} information. In this work, GSM is used as one of the two-dimensional maps.
**Algorithm 1:** Estimation of SM and GSM:

1. Get the input document image, its rough binarization, and a priori information including $w_s$ and $h_l$;
2. Estimate the edge map of the document image at the $h_l$ scale (using Sobel’s method [37], and mosaicking the image with squares of size $h_l \times h_l$);
3. Estimate the skeleton map of the document image using the thinning method;
4. Ignore the skeleton whose their distance to the edges is more than $\lfloor w_s/2 \rfloor + 1$;
5. Produce the Euclidean distance map of the skeleton map capped to the $w_s + 1$;
6. Estimate the text edges based on the calculated distance map (by selecting those pixels which have a value of $w_s + 1$ on the distance map and also are presented on the edge map of step 2);
7. Produce the Euclidean distance map of the new edge map capped to the $w_s + 1$;
8. Combine the two distance maps, obtained in steps 5 and 7, to generate the gray stroke map (GSM);
9. Threshold the GSM on 0.5 to generate the stroke map (SM);

Fig. 4. Illustration of SM performance. a) input image suffering from degraded background and synthetic degradation. b) rough binarization of the (a) using Otsu’s method. c) and d) GSM and SM of (b) generated using Algorithm 1.
Fig. 5. The detailed steps used to produce the outputs in Figure 4. a) The edge map calculated in the step 2 of Algorithm 1. b) The skeleton map calculated in the step 3. c) The corrected skeleton map calculated in the step 4. d) The Euclidean distance map of the skeleton map as calculated in the step 5. e) The estimated text edges calculated in the step 6. f) The Euclidean distance map of the new edge map as calculated in the step 7.
5 Experimental results

For the experiment, a database of 2,400 images of 60 unique subwords is selected from a real Arabic manuscript which was used in the construction of the IBN SINA database [14]. It is worth noting that the number of binary descriptors is independent from the number of sample. For example, if we just need to consider the descriptors which count the presence of a character in a letter block, we will have only 28 binary descriptors presented. However, in this work, because of small size of the database, instead of the binary descriptor, the actual labels of the 60 samples are considered as the binary labels to be learned. We will use larger databases along with actual binary descriptors in the future work. In Figure 6, the histogram of the number of letters in the database letter blocks is shown. The subword images are resized using padding; in the first step (resizing step), the image with the maximum width is selected and the other images are resized by adding zero/one columns (based on the background being zero/one) to the left or right side of the images. Next, the image with the maximum height is selected, and the other images are resized by adding zero/one row to the top or bottom of the image based on the baseline. Because Arabic is written from right to left, we add zero columns to the left side of the images. Each image is converted to a feature vector of $1 \times N$, and the database is created by concatenating all the feature vectors. In our case, each feature vector has 13,480 elements.

For each experiment, we use two cases to create the database: i) binary images (BIN) and ii) binary and GSM images (BIN+GSM). This means that, for each sample, we use the binary image and convert it to a feature vector in one case, and we use the binary image and the GSM image and convert them to a vector
in the second case, creating the final feature vector by concatenating them.

5.1 PCA

in the first experiment, PCA is applied for dimension reduction, and then a K-NN classifier is applied. The result constitutes our baseline for comparing performances. In the first step, the mean and covariance matrices of the features are computed, and the principal components are calculated using eigenvalue decomposition of the covariance matrix. The eigenvalue values are sorted and the \( n \) maximum values are selected. The corresponding eigenvectors are used for the projection. The value of \( n \) can be selected based on the energy of the components. For example, this value could be selected so that 90% of the energy is preserved. In our database, the data dimension is 13,480 and we kept 1,200 eigenvectors for the projection in order to preserve more than 99% of the energy. The K-NN classifier is first applied on training and verification data to select the optimal \( K \), which is then used in the testing step. Table 1 shows the result of applying the first experiment with the two types of data, BIN and BIN+GSM. It is worth noting that, in the second case, in which GSM is used in the input representations, the performance is improved by 3%.

Fig. 6. Histogram of the image letter blocks based on label length.
<table>
<thead>
<tr>
<th>DATA type</th>
<th>k</th>
<th>Training set (%)</th>
<th>Testing set (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Binary</td>
<td>1</td>
<td>76.8</td>
<td>72.2</td>
</tr>
<tr>
<td>Binary + GSM</td>
<td>1</td>
<td>80.1</td>
<td>75.9</td>
</tr>
</tbody>
</table>

Table 1

The Experimental results of the PCA and K-NN classification case.

<table>
<thead>
<tr>
<th>DATA type</th>
<th>Dimensions</th>
<th>number of trees</th>
<th>Testing set (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Binary</td>
<td>1200</td>
<td>460</td>
<td>78.61</td>
</tr>
<tr>
<td>Binary</td>
<td>256</td>
<td>420</td>
<td>81.25</td>
</tr>
<tr>
<td>Binary + GSM</td>
<td>1200</td>
<td>440</td>
<td>80.33</td>
</tr>
<tr>
<td>Binary + GSM</td>
<td>256</td>
<td>440</td>
<td>82.97</td>
</tr>
</tbody>
</table>

Table 2

The Experimental results of the PCA and RF classification case.

Also, as a second baseline, the ensemble learning using Random Forest technique [9] is applied to the database. The results are presented in Table 2.

5.2 **PCA+GDA**

In the second experiment, we use the GDA algorithm in the dimension reduction step, in addition to PCA. The RBF kernel is used in the GDA. In order to select the best value for the parameter $\sigma$, the data is divided into two parts: training and verification. At each step, the GDA model is created using the training database and the parameter is selected, and then we test the model using the verification set. We select the parameter that maximizes the rate of
Table 3
The Experimental results of the PCA+GDA approach.

classification. In Table 3, the results of the second experiment are shown. The results are obtained using the RBF kernel with $\sigma = 11$. As can be seen from the results, the application of manifold learning has increased the performance by 5%.

5.3 PCA+LLE

In the third experiment, we use the LLE algorithm as our dimensionality reduction approach. The Euclidean distances between all the samples are calculated and the distance matrix is modified by applying the distance that is used in SLLE, the distance (4). After this step, the samples with the same labels become closer, and the samples with the different labels grow farther apart. We applied a clustering algorithm to divide the samples into C clusters, as shown in Figure 7. In this experiment, we know the number of classes; therefore, we set a value of 60 to the number of clusters.

Using the modified distance matrix, we expect the samples with the same labels to be clustered. A manifold for each cluster (we have 60 clusters here) is created using the LLE algorithm. The Euclidean distance of the samples is used to select the $K$ nearest neighbors, and then the weights of reconstruction of each sample by its neighbors are calculated. The eigenvalues and eigenvec-
Fig. 7. Using label-dependent distance, the data are separated into several manifolds, one for each class.

tors of the cost matrix $M = (I - W)'(I - W)$ are calculated, and the samples are projected into a low-dimensional space constructed using $d$ eigenvectors, corresponding to the $d$ smallest eigenvalues, $d = K - 1$ in this experiment. The quality of the projection can be analyzed by some criterion, such as (5) or (6). In this experiment, we selected the second criterion, which checks whether or not the local property of the data in the original space is preserved correctly in the projected space. Ideally, the samples should have the same nearest neighbors in the original and projected spaces. This means that samples which are close in the original space should be close in the projected space, and samples which are far apart in the original space should be far apart in the projected space. $K$ is the only parameter that exists in LLE that affects the property mentioned above. So, the optimum value of $K$ can be obtained by changing the value of $K$ and projecting the data, and then measuring the manifold quality for achieving the optimum value that minimizes the criterion. The samples in the training set are used to optimize the parameter, and the samples in the testing set are used to measure the performance of the algorithm. Optimal $K$
values for some of the manifolds are presented in Table 4. The learning steps of this experiment are as follows:

- Select an initial value for $K$.
- Create a manifold (projecting the samples) using the selected $K$.
- Measure the quality of the manifold by means of criterion (6).
- Change the value of $K$ and repeat the 2 previous steps.
- Find the optimum value for $K$.
- Repeat the previous steps for each cluster to find the best parameter.

In the testing procedure, each new piece of data should be projected onto all the manifolds, so that the decision will be the label of the new data, which is the label of one of these manifolds. In the learning step, as we create each manifold separately and because the optimum values of $K$ are different for each manifold, the direct combination of the result of the projection onto different manifolds is not possible. The idea behind LLE is to project data into the new space while preserving the local properties of the data, and this is achieved by finding the weights of reconstruction of one sample by its nearest neighbors. So, if the new data are projected into the proper manifold, there should be a the minimum of reconstruction errors. The testing steps are as follows:

- Find the K-nearest neighbors of a new sample, $x_t$, on each manifold, say the $i^{th}$ manifold, using the Euclidean distance and the optimum value of $K$ for that manifold.
- Find the reconstruction weights in the $i^{th}$ manifold: $w_j$.
- Calculate the reconstruction error in the $i^{th}$ manifold:

$$e_i(x_t) = \|x_t - \sum_j w_j x_j\|$$  \hspace{1cm} (23)
For the new sample, select the manifold that produces the smallest amount of reconstruction error:

\[ \tilde{i} = \arg \min_i e_i(x_i) \] (24)

The results of this experiment are shown in Table 5. Again, it can be seen that LLE improves performance by 3%.

6 Conclusion and future prospects

With the introduction of multi-class classification techniques [18, 19, 20] and the introduction of alternative class-reduction approaches, such as equivalent binary problem technique [14], performing recognition at the letter block (sub-word) level is feasible. In this work, an objective approach to the recognition of letter blocks based on their image representation is introduced. Raw data from the binary images is learned using two manifold learning techniques. In order to reduce the computation complexity of the manifold learning step, the very high dimensional input representations are first reduced using PCA. The performance of the system is then increased by adding a stroke membership map.
Table 4

Optimum value of $K$ for some sample subwords. The labels are Fingilish transliterations [14].

<table>
<thead>
<tr>
<th>Subword image</th>
<th>Labels</th>
<th>Optimum value of $(K)$</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>bnv</td>
<td>18</td>
</tr>
<tr>
<td></td>
<td>n</td>
<td>16</td>
</tr>
<tr>
<td></td>
<td>a</td>
<td>12</td>
</tr>
<tr>
<td></td>
<td>ld</td>
<td>14</td>
</tr>
<tr>
<td></td>
<td>lmqa</td>
<td>16</td>
</tr>
</tbody>
</table>

Table 5

The experimental results of the PCA+SLLE approach.

<table>
<thead>
<tr>
<th>DATA type</th>
<th>Testing set (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1 Binary</td>
<td>76.8</td>
</tr>
<tr>
<td>2 Binary + GSM</td>
<td>79.1</td>
</tr>
</tbody>
</table>
(GSM) to the binary images to represent the input of each letter block. The performance of the proposed approaches has been tested on a database from a historical Arabic manuscript with promising results. It has been observed that including GSM in the input representation and independently applying manifold learning improves performance by 3% and 5% respectively.

In future work, the highly multi-class nature of recognition at the letter block (subword) level will be addressed using various approaches, including the binary descriptors approach [14, 15]. In another direction, the problem of greater sensitivity to stroke variations introduced by the pixel-level representation will be addressed using nonlinear transformations on the binary images.

The authors thank the NSERC of Canada and the SSHRC of Canada (Indian Ocean World MCRI Project) for their financial support.

References


[5] Mauro Barni, Jean-Angelo Beraldin, Christian Lahanier, and Alessan-


Jose, CA, USA, October 14–16 2009.


[39] Juliana Valencia-Aguirre, Andrés Álvarez Mesa, Genaro Daza-


