Handwritten digit recognition of Indian scripts: a cascade of distances approach

Hubert Cecotti

School of Computing and Intelligent Systems
Ulster University
Londonderry, Northern Ireland, UK
email: h.cecotti@ulster.ac.uk

Abstract—The recognition of handwritten digits remains a difficult problem, particularly in some scripts where there exists a large variation of style across writers. This large variability is an interesting challenge for algorithms in image processing and pattern recognition. Thanks to the accelerating progress and availability of low cost computers, high speed networks, and software for high performance distributed computing, it is possible to use computational expensive technique on large databases. In this paper, we propose to investigate the impact on the accuracy of different parameters and pre-processing methods of a distance based on image distortion models. A key challenge is to reduce the processing time of the nearest neighbor classification by considering rejection rules and adaptive distances. We propose to evaluate the performance of single character recognition on three databases of Indian handwritten digits, each database corresponds to a popular Indian script: Bangla, Devnagari, and Oriya. We show that the extraction of features related to four directions allows a significant improvement of the accuracy. The proposed approach takes advantages of GPU and high performance clusters, providing state-of-the-art performances.

I. INTRODUCTION

The evolution of pattern recognition techniques over the years has taken advantage of both theoretical and hardware progresses. For instance, classifiers based on neural networks have exploited both unsupervised pre-training [1], and graphics processing unit (GPU) [2]. The corresponding improvements given by theoretical and hardware progresses invite to perpetually reconsider the methods to use for classification. Particularly, research in neural networks for classification tasks has progressed thanks to deep learning [3]. Moreover, while convolutional neural networks have provided state-of-the-art performance since their introduction [4], the use of GPUs has significantly increased their interest. One of the key challenges in machine learning techniques applied on large databases is to determine the best trade-off between the efficiency, the possibility to implement the method on computer clusters and/or GPU, and the possibility to update the system incrementally with new data. The same principle can be extended to image processing techniques, which can also benefit from computer clusters and GPUs to process large databases of images (*e.g.,* natural images, handwritten characters). Therefore, it is fundamental to obtain methods that have not only a high accuracy, but that can be take advantage of a parallel implementation to be

computationally efficient. For the Latin script, the recognition of single handwritten characters (offline and online) has a sufficient high accuracy to claim that this research problem is almost over [5], [6]. This high performance is due to efficient machine learning and feature extraction techniques. The problem of isolated character recognition is dominated by deep learning architecture such as convolutional neural networks [7], and Support Vector Machines (SVM) [8], or by combining both approaches [9], [10]. However, the accuracy of single handwritten character recognition remains below 100% in some scripts because documents are not properly conserved, and are therefore noisy. Furthermore, there exists a large variability across writers, with different glyphs for a same digit or character. This is why it remains important to still propose new methods to reach a perfect score. In addition, there always exist documents with noisy and/or deformed characters, which cannot be recognized with current optical character recognition (OCR) technologies [11]. The recognition of some characters can be impossible without any contextual information. Depending on the application, the character recognition procedure can be followed by other post-processing steps, which include a language model (*e.g.,* a lexicon, a grammar).

In this study, we recall the advantage of k-nearest neighbor for parallel implementation, and we propose a rigorous evaluation of Image Distortion Model Distance (IDMD), and its variation on three databases on single handwritten digits from representative Indian scripts. While IDMD is an efficient distance [12], its computation time remains important. This paper explores solutions to reduce the number of images processed by a time consuming distance, and the number of prototypes for the comparisons. To solve this problem, we propose rejection rules to select a limited number of images to test, in order to decrease the computation time while keeping a high accuracy of the non-rejected images. The remainder of the paper is organized as follows: First, we give an overview of the k-nearest neighbor and image matching techniques in character recognition. Then, we present a solution based on a sequential combination of classifiers. In Section IV, we present the three handwritten databases of Indian digits. The different distances and their combination are then evaluated in Section V. Finally, the performance of the proposed approach is discussed in Section VI.

II. RELATED WORK

A. K-nearest neighbor

K-nearest neighbor (k-nn) technique is a well studied and fundamental method in many computer vision and character recognition systems. Given two databases D_1 and D_2 that contain N_1 and N_2 examples. Each example is described by a couple $(x_j(i), y_j(i))$, $1 \leq i \leq N_j$, $1 \leq j \leq 2$. $x_j(i)$ corresponds to a set of N_f features, $x_j(i) \in \mathbb{R}^{N_f}$, and y represents the associated label $y \in \{1..N_c\}$, where N_c is the number of classes in the problem. For classification, D_1 and D_2 correspond to the training and test databases, respectively. The k-nearest neighbor classifier estimates the labels of each record of D_2 by using the k nearest neighbors from D_1. The probability of error of this classifier is bounded above by twice the Bayes probability of error [13]. k-nn is a search problem that is often used as a baseline for machine-learning classification algorithms, computer vision, and data mining applications because of its simplicity. k-nn classifiers involve the choice of the parameter k, and a relevant distance for the comparisons of the examples in the databases. In pattern recognition, k-nn is one of the oldest methods, and it is often regarded as an inefficient classifier compared to more recent classifiers such as Support Vector Machines or artificial neural networks due to two main problems: first, the inability to obtain an efficient distance for the classification of images, second the high computation cost related to the number of prototypes to compare, and the distance itself. Hundreds of studies have tackled different issues of k-nn, the computational time problem can be managed by considering appropriate data structures (*e.g.,* KD-trees [14]).

Given the increasing volume of data, and the required computation time for evaluating the distance between two objects, it is difficult to perform efficiently k-nn on a centralized machine. Fortunately, k-nn can be easily transferred into a parallel implementation. Both the training database and the test database can be cut into different blocks that can be processed independently because the main goal is to compute the distance between all the possible couples of training/test patterns ($N_1 \times N_2$ distances are computed). This can be achieved with a shared-nothing cluster on a number of commodity machines using MapReduce, which is a simple and powerful parallel and distributed computing paradigm [15]. It is worth noting that novel algorithms have been proposed in MapReduce to perform efficient parallel k-nn joins on large data [16], [17].

B. Image matching

In handwritten character recognition, image matching techniques can be regarded as inefficient techniques due to their high processing time. Elastic matching techniques can be classified into two categories: parametric and non-parametric [18]. It is typically seen as an optimization problem of two-dimensional warping (2DW).

This problem is directly related to point matching, which has to deal with the existence of outliers and geometric transformations that may require high dimensional non-rigid

mappings [19]. Deformations in handwritten characters can be of two types: first, the global or large deformations such as rotation (with limited angles), scaling, translation; and second, the local deformations that include changes of stroke direction, curvature, and length of the lines. The local deformations depend on the pen/pencil that is used, as the thickness of the lines will depend on what is used to write a character. Due to the different type of deformations that can occur within the same character, it is difficult to determine models of deformations.

Keysers et al. presented an efficient distance for the classification of handwritten digits based on image distortion models [12]. Moreover, they determined that more complex models (*e.g.,* 2-dimensional warping) do not necessarily represent better models compared to the simple image distortion model. Despite the high accuracy on MNIST (error rate=0.54%), and without extending the size of the training database, this distance has a high computational cost when dealing with large databases. To reduce the computational cost to process a database, a strategy consists in limiting the number of images to process. In addition, it is also possible to reduce the number of prototypes that should be used during the comparisons.

We define the distance L_p between two images I_1 and I_2 of size $N_s \times N_s$ by:

$$L_p = \left(\sum_{i=1}^{N_s} \sum_{j=1}^{N_s} |(I_1(i,j) - I_2(i,j))|^p \right)^{1/p} \quad (1)$$

The Weighted Euclidean distance is defined by:

$$L_{w2} = \sqrt{ \sum_{i=1}^{N_s} \sum_{j=1}^{N_s} \frac{1}{s^2(i,j)} (I_1(i,j) - I_2(i,j))^2 } \quad (2)$$

where $s^2(i,j)$ is the variance of the point (i,j).

The image distortion model distance (IDMD) takes as input two images I_1 and I_2 of size $N_s \times N_s$. Due to the use of the filters and the displacement fields, the image has to be placed in a larger image with a border of the background color to include the possible shifts. The distance is then computed through a range of pixels from N_{min} to N_{max}. The IDMD is calculated as follows:

$$\text{IDMD} = \sum_{i_1=N_{min}}^{N_{max}} \sum_{j_1=N_{min}}^{N_{max}} d_1(i_1, j_1) \quad (3)$$

where

$$d_1(i_1, j_1) = \min_{(i_2, j_2) \in \{-w0;w0\}^2} d_2(i_1, j_1, i_2, j_2) \quad (4)$$

$$d_2(i_1, j_1, i_2, j_2) = \sum_{i_3=-w_1}^{w_1} \sum_{j_3=-w_1}^{w_1} \sum_{i_4=1}^{w_2} |v_1 - v_2|^p \quad (5)$$

where v_1 and v_2 are the pixel values at the following coordinates in the i_4 pre-processed image ($1 \leq i_4 \leq w_2$):

$$v_1 = I_1(i_1 + i_3, j_1 + j_3) \quad (6)$$

$$v_2 = I_2(i_1 + i_2 + i_3, j_1 + j_2 + j_3) \quad (7)$$

For each pixel (i_1, j_1), we use a displacement field of size w_0 in each direction (it corresponds to the elements of a window of size $2w_0 + 1$), a window for the consideration of the neighborhood pixels of size $2w_1 + 1$, and the sum of w_2 values, which corresponds to w_2 filtered images. It is worth noting that the *min* function aims at including a relative invariance to local deformations. The purpose of this procedure is similar to what is achieved through layers with a pooling function, *i.e., max* function in convolutional neural network [20]. With Sobel filters on the horizontal and vertical direction, $w_2 = 2$. The Euclidean distance (L_2) between I_1 and I_2 is similar to IDMD with the following parameters: $w_0 = 0$, $w_1 = 0$, $w_2 = 1$, $p = 2$.

In the next sections, we consider the following parameters for IDMD: $w_0 = 2$, $w_1 = 1$, and $p = 2$. We evaluate three pre-processing procedures: First, by considering only the pixel value of the image ($w_2 = 1$), then by using Canny descriptors of the image [21], and third, by using images after Sobel filtering. We consider two sets of filters: horizontal and vertical filters only ($w_2 = 2$), and with the diagonals ($w_2 = 4$). We also consider the method $L_2^{Sobel(4)}$ that considers the distance L_2 with inputs after Sobel filtering (f_1, f_2, f_3, and f_4), and after decreasing the size of the resulting images by two. Hence, the number of features with $L_2^{Sobel(4)}$ is the same as in L_2. The filters are precised thereafter:

$$f_1 = \begin{bmatrix} 1 & 0 & -1 \\ 2 & 0 & -2 \\ 1 & 0 & -1 \end{bmatrix} \quad f_2 = \begin{bmatrix} 1 & 2 & 1 \\ 0 & 0 & 0 \\ -1 & -2 & -1 \end{bmatrix} \quad (8)$$

$$f_3 = \begin{bmatrix} 0 & 1 & 2 \\ -1 & 0 & 1 \\ -2 & -1 & 0 \end{bmatrix} \quad f_4 = \begin{bmatrix} 2 & 1 & 0 \\ 1 & 0 & -1 \\ 0 & -1 & -2 \end{bmatrix} \quad (9)$$

III. HIERARCHICAL PROCESSING

Multi-classifier systems can be used to optimize the performance by combining the decision of several classifiers [22], [23], [24]. Multi-classifiers with sequential architectures in multi-class classification are typically used with general classifiers at the top of the chain, and more specialized classifiers in the next steps. This type of combination can be used to reduce the number of classes to process, *i.e.,* the classifier provides as output a subset of potential classes, then a more dedicated classifier is used for the classification of those subset of classes [25]. In the proposed system, the classifier combination is justified by the computational cost that is needed to classify an image. While it would be possible to use the most reliable distance for the evaluation of the whole database, its computational cost may not allow the process of high volumes of images. We consider a cascade of k-nn classifiers based on two distances. The goal of this classifier combination is to reduce the processing time for the evaluation of a database of images. A simple method is considered as a first step, then a more computational expensive method is used for the patterns that could not reach a strong decision at the previous level.

The problem is to find a rejection rule that limits the number of images while keeping a high accuracy for the recognized images. First, we compute k-nn with the distance L_2 and $k=500$, for the selection of the prototypes that will be used in the next step. The evaluation of k-nn with distance L_2 and $k=10$, which is based on the distances computed for $k=500$, includes a rejection rule: all the k best answers must belong to the same class in order to accept the decision. If it is not the case, then the image is evaluated with IDMD, where the training prototypes are the 500 best answers obtained with distance L_2 at the previous stage, with $k=3$. At the second level, images can be rejected with the same decision rule as in the first level: all the k best answers must belong to the same class. The proposed approach is depicted in Figure 1.
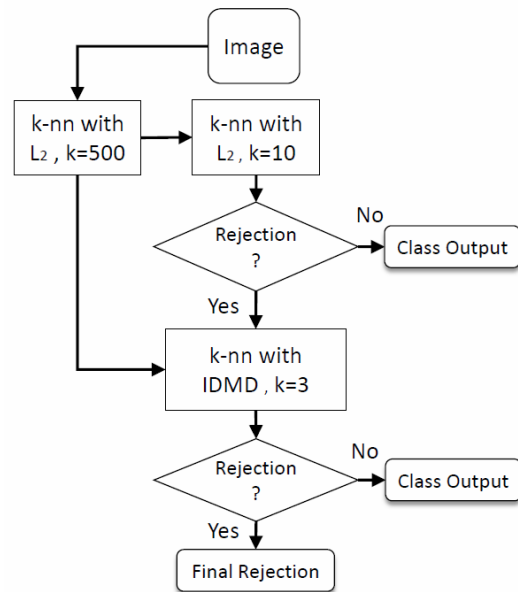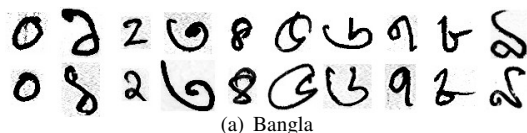

Fig. 1. Multi-classifier system overview.

IV. DATABASES

Three databases of handwritten digits have been chosen for the analysis of the performance. The three databases contain images of digits in three Indian scripts: Bangla, Devnagari, and Oriya. Samples of digits are presented in Figure 2. All the images were normalized with the same procedure. Because some databases have very noisy images and/or images in color, images were first binarized with the Otsu method at their original size [26], then they were size normalized to fit in a 20x20 pixel box while preserving their aspect ratio. The resulting images contain 8 bit gray levels due to the bicubic interpolation for resizing the images. Finally, all the images were centered in a 28x28 pixel box field by computing the center of mass of the pixels, and translating the gravity center of the image to the center of the 28x28 field. Table I presents for each database the number of classes, the total number of images in the database, and
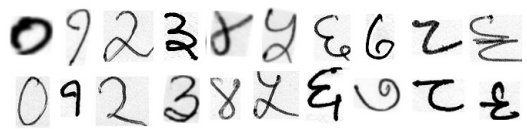
Database	Bangla	Devnagari	Oriya
# classes	10	10	10
	Training		
# samples	19392	18783	4970
# per class	360	1878 ± 15	497 ± 3
size (x)	58 ± 16	65 ± 16	73 ± 25
size (y)	54 ± 16	62 ± 19	73 ± 26
	Test		
# samples	4000	3763	1000
# per class	400	376 ± 3	100
size (x)	59 ± 17	66 ± 17	75 ± 25
size (y)	54 ± 18	62 ± 20	74 ± 26

| D	k	Bangla	Devnagari	Oriya
L_1	1	6.75	3.96	9.40
	3	6.68	4.04	9.30
	5	6.70	3.93	7.70
	10	7.68	4.54	8.10
L_2	1	6.53	3.91	8.80
	3	6.30	3.75	8.00
	5	6.45	3.61	**7.50**
	10	7.28	4.17	7.90
L_3	1	6.58	4.07	9.20
	3	**6.00**	3.72	9.10
	5	6.38	3.67	8.40
	10	7.05	4.36	8.30
L_{w2}	1	11.90	9.15	6.70
	3	11.30	8.97	7.39
	5	10.60	9.12	7.89
	10	10.90	9.55	8.03
$L_w^{Sobel(4)}$	1	6.37	3.70	8.80
	3	11.45	**3.51**	8.20
	5	11.49	3.67	8.00
	10	12.08	4.62	8.40

the number of images per class, for both training and the test.

(a) Bangla

(b) Devnagari
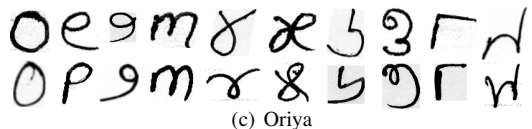
(c) Oriya

Fig. 2. Representative handwritten digits for the different databases (from zero to nine).

A. Indian handwritten digits

India is a multilingual country, with twenty two official languages and twelve scripts. In Indian language scripts, the concept of upper case and lower-case characters is not present. The databases of Indian digits were created at the Indian Statistical Institute, Kolkata, India [27], [28], [29]. The databases contain Bangla digits. Bangla is the fourth most popular script in the world, used by more than 200 million people [30], [31]. The second database corresponds to Devnagari digits, which is part of the Brahmic family of scripts of India, Nepal, Tibet, and South-East Asia [32]. Devnagiri script is used to write Hindi, Konkani, Marathi, Nepali, Sanskrit, Bodo, Dogri and Mathili. The third database contains Oriya (Utkala Lipi) digits [33]. This script is one of the many descendants of the Brahmi script of ancient India. In [33], Bhowmik et al. obtain an accuracy of 90.50% by using Hidden Markov Models. In [34], they propose a two-stage framework that combines modified quadratic discriminant function (MQDF) [35] and MLPs for the recognition of Bangla characters. On other databases of the Bangla characters (50 classes), Mandal et al. use features based on the combination of gradient features and Haar wavelet

coefficients at different scales with a k-nn classifier to reach an accuracy of 88.95% [36]. In [37], Bhattacharya et al. propose a method based on a chain code histogram feature with an MLP classifier, and obtain an accuracy of 88.95%.

V. RESULTS

A. Distance evaluation

The error rate for the three databases, by using only the distances L_1, L_2, L_3, L_{w2}, and $L_2^{Sobel(4)}$ is presented in Table II; the best results for each database are in bold. The input of the k-nn classifiers are the images in gray level of size 28x28.

The performance obtained with only IDMD is presented in Table III. The parameters $w_0 = 2$ and $w_1 = 1$ were estimated on the training database of the Bangla database by comparing the IDMD score between each image and its closest neighbor of the same class. The worst performance is typically obtained with k=10. In the following sections, we consider k=3. The best performance is obtained with IDMD by using images filtered with Sobel in four directions. The error rate is 1.70, 0.77, and 2.00 for Bangla, Devnagari, and Oriya, respectively.

The error rate decreases with the addition of more prototypes with k-nn (k=3). The speed of the classification is linearly dependent on the number of prototypes used during the comparisons. For the three databases, we observe the same pattern of performance with an error decreasing significantly with the increase of prototypes, until 150, then it reaches a plateau where the error decreases slowly. For instance, the error rate for the Bangla digits decreases only by about 0.2% from 100 to 500 prototypes used in the comparisons, *i.e.,* by multiplying the processing time by five. The evolution of the error rate in relation to the rejection rate of k-nn based on a consensus decision is depicted in Figure 3 for L_2, and Figure 4 for IDMD_{sobel}. Each point in the curves correspond to the choice of k in the decision.

TABLE III
ERROR RATE (IN %) WITH ONLY IDMD ($w_0 = 2, w_1 = 1$), AND
CONSIDERING THE 500 BEST PROTOTYPES FROM K-NN WITH L_2.

Input	k	Bangla	Devnagari	Oriya
	1	2.08	1.01	2.70
Pixel	3	2.20	0.82	3.50
	5	2.20	0.82	3.50
	10	2.78	1.44	4.00
	1	2.08	1.36	3.20
Canny	3	2.28	1.30	3.10
	5	2.08	1.41	3.30
	10	2.55	1.89	4.10
Sobel	1	1.45	0.88	2.00
(2 dir.)	3	1.72	0.80	2.10
	5	1.77	0.80	2.50
	10	2.77	1.01	3.40
	1	**1.45**	0.85	**1.90**
Sobel	3	1.70	**0.77**	2.00
(4 dir.)	5	1.88	0.77	2.30
	10	2.05	1.12	3.40

TABLE IV
CONFUSION MATRIX FOR THE BANGLA DATABASE WITH IDMD$_{sobel}$ (4
DIRECTIONS) ($w_0 = 2, w_1 = 1, k = 1$), AND CONSIDERING THE 500
BEST PROTOTYPES FROM K-NN WITH L_2.

	0	1	2	3	4	5	6	7	8	9
0	399	1	0	0	0	0	0	0	0	0
1	0	378	1	0	12	0	0	0	0	9
2	0	0	400	0	0	0	0	0	0	0
3	0	0	1	397	0	0	2	0	0	0
4	0	0	0	0	399	0	0	0	1	0
5	0	0	0	0	2	397	1	0	0	0
6	0	0	0	1	0	6	393	0	0	0
7	0	0	0	0	0	0	0	400	0	0
8	0	0	0	0	0	0	0	0	400	0
9	0	17	0	0	2	1	0	1	0	379

The confusion matrix for IDMD$_{sobel}$ (4 directions) are given in Tables IV, V, and VI.

B. Rejection and subset of prototypes

The results corresponding to the cascade of decisions between L_2 and IDMD$_{sobel}$ are presented in Table VII. First, despite a high threshold for the decision, there remains an error rate of 0.28, 0.14, and 0.65 with the distance L_2 on Bangla, Devnagari, and Oriya databases. With the rejection option of the first level, about 30% of the images are rejected to be processed by the second level. In the second level, which is supposed to process the most difficult images,

TABLE V
CONFUSION MATRIX FOR THE DEVNAGARI DATABASE WITH IDMD$_{sobel}$
(4 DIRECTIONS) ($w_0 = 2, w_1 = 1, k = 1$), AND CONSIDERING THE 500
BEST PROTOTYPES FROM K-NN WITH L_2.

	0	1	2	3	4	5	6	7	8	9
0	369	0	0	0	0	0	0	0	0	0
1	0	372	1	1	2	1	0	1	0	0
2	0	1	368	4	0	3	0	0	2	0
3	0	0	2	373	1	0	0	1	0	0
4	1	0	0	0	373	1	0	1	0	0
5	0	1	0	0	1	376	0	0	0	0
6	0	0	1	0	0	0	373	0	0	0
7	2	0	0	0	0	0	1	375	0	0
8	0	0	0	0	0	0	0	0	376	1
9	0	0	0	0	0	0	1	0	1	376

TABLE VI
CONFUSION MATRIX FOR THE ORIYA DATABASE WITH IDMD$_{sobel}$ (4
DIRECTIONS) ($w_0 = 2, w_1 = 1, k = 1$), AND CONSIDERING THE 500
BEST PROTOTYPES FROM K-NN WITH L_2.

	0	1	2	3	4	5	6	7	8	9
0	100	0	0	0	0	0	0	0	0	0
1	0	99	1	0	0	0	0	0	0	0
2	0	0	99	0	0	0	0	1	0	0
3	0	0	0	99	0	0	0	1	0	0
4	0	0	0	0	100	0	0	0	0	0
5	0	0	0	0	0	100	0	0	0	0
6	0	0	3	0	0	0	92	5	0	0
7	0	0	7	0	0	0	0	93	0	0
8	1	0	0	0	0	0	0	0	99	0
9	0	0	0	0	0	0	0	0	0	100

TABLE VII
PERFORMANCE (IN %) FOR THE COMBINATION OF L_2 (LEVEL 1) AND
IDMD$_{sobel}$ (LEVEL 2), ($w_0 = 2, w_1 = 1$), WITH AND WITHOUT
REJECTION IN THE SECOND LEVEL.

		Bangla	Devnagari	Oriya
	level-1 error	0.28	0.14	0.65
	level-1 rejection	29.92	24.10	38.30
Without	level-2 error	5.35	2.98	4.96
rejection	total error	1.80	0.82	2.30
With	level-2 error	1.84	0.83	1.75
rejection	level-2 rejection	9.36	6.72	10.70
	total error	0.72	0.30	1.04
	total rejection	2.80	1.62	4.10

the error rate is 1.80, 0.82 and 2.30. When we consider the possibility to reject images in the last stage of the combination (level 2), an error rate of 0.72, 0.30 and 1.04 is obtained with a rejection rate of 2.80, 1.62, and 4.10, for the Bangla, Devnagari, and Oriya databases.

C. Processing time

The system was implemented with Matlab R2013a. With a parallel implementation on three GPU cards (NVIDIA Tesla C1060), it takes about 26, 26, and 5 seconds to get the distances for k-nn with the Euclidean distance and k=500, for processing the Bangla, Devnagari and Oriya test database. With a parallel implementation on a cluster using 50 cores (Intel Xeon X5650 2.66 Ghz), it takes about 406, 410, and 105 additional seconds to process the Bangla, Devnagari and Oriya test database with IDMD$_{sobel}$, 500 prototypes, and k=3. The processing time is linearly dependent on the number of examples in the training and test databases. By applying the rejection rule at only the second level, the current implementation allows us to process the Bangla, Devnagari and Oriya databases in about 143, 124, and 45 seconds with state-of-the art performance.

VI. DISCUSSION

The main goal of the proposed method was to propose an efficient combination of distances that can be easily implemented on a computer cluster, and that can provide state-of-the-art performance in handwritten digit recognition on several databases. The problem was to determine an efficient combination of distances for handwritten digits from several Indian scripts with a minimum number of hyper-parameters, and without prior knowledge from the script.
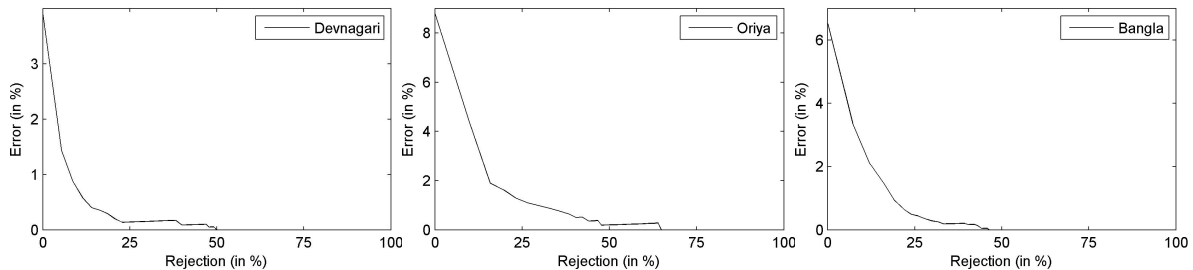
Fig. 3. Evolution of the error rate in relation to the rejection rate based on a consensus decision given by k, with L_2.
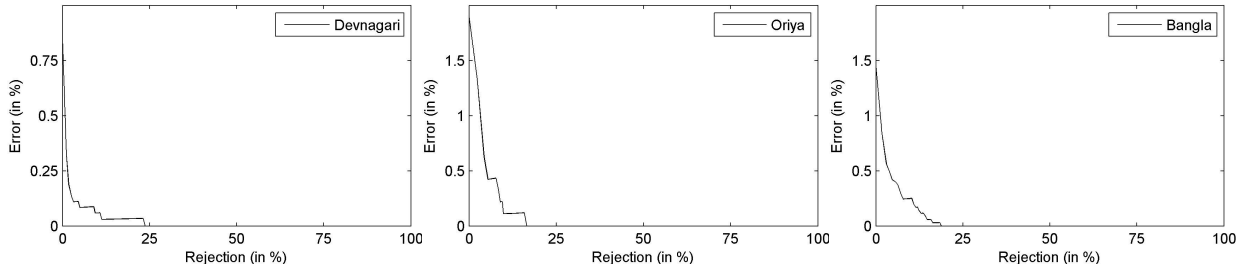

Fig. 4. Evolution of the error rate in relation to the rejection rate based on a consensus decision given by k, with $IDMD_{sobel}$.

Whereas methods such as convolutional neural networks propose the best results in MNIST [7], and other databases of images and signals [38], it can be difficult to determine the ideal architecture of the network without a prior expertise of the problem. The description of the architecture includes the number of layers, the number of neurons in each hidden layer, the number of maps per layer, the choice of activation functions, and the right combination of pooling, subsampling and convolution between hidden layers. Furthermore, the number of classifiers shall be determined for systems that include the decision from a committee of classifiers. For the addition of elastic deformations, some knowledge of the script must be assumed: the parameters of the deformations, *e.g.,* intensity, parameters of the Gaussian for filtering the random fields, must be carefully chosen. For Indian scripts, this parameter is difficult to set because the variation across writers is large. With IDMD, only three main parameters have to be chosen: the size of the possible shifts (w_0), the size of the neighborhood of each pixel (w_1), and the number of pre-processed images (w_2). We have shown that the latter parameter is relevant as the filtering method plays an important role in the accuracy. Moreover, features using Sobel with four directions provided a better accuracy than features using Sobel with only two directions.

Since IDMD is computationally expensive, the recognition system must take advantage of parallel computing and computer clusters. A cascade of classifiers based on the computation times of each distance has been proposed, allowing a focus on difficult images. The speedup can be increased by both the number of computers, and the rejection rule that limits the number of images to process with IDMD. The evolution of parallel computing enables a shift of paradigm for how machine learning and pattern recognition

techniques should be employed. Image processing techniques that may only be used for natural image retrieval due to their computational costs may be used for handwritten character recognition thanks to parallel computing. IDMD allows the invariance to local deformation but it remains sensitive to transformations such as dilatation, and large rotations. Since the possible deformations are dependent of the script, additional information could be added in IDMD. For instance, w_0 could be set in relation to the distance to the center of the images because points that are far from the center are more likely to be displaced at a further distance than points that are close to the center for rotated images. Finally, deformed patterns can be added in the databases to increase the size of the training database, and therefore increase the ability of the classifier to generalize, but the parameters of the transformations have to be tuned in relation to the script. Furthermore, the introduction of images with important deformations may involve the introduction of examples that do not correspond to the right class for Indian scripts. Finally, the database of Indian digits is very noisy, and may require better denoising techniques.

VII. CONCLUSION

A comprehensive hierarchical combination of distances between gray level images of handwritten characters has been presented in this paper. This combination allows to obtain a relatively fast estimation of the performance for handwritten character recognition. While classifiers based on k-nearest neighbor can be considered as classifiers with poor performance for handwritten characters due to the use of the common L_2 distance, we have proven that the method, with appropriate distances, is relevant for the recognition of handwritten digits in different scripts. By combining the advantages of parallel computing such as high performance

clusters and GPUs, and rejection rules, it is possible to significantly increase the speed of the approach, allowing the use of distances that are computationally expensive, such as the Image Deformation Model Distance. As there are twelve scripts in India, further works will be carried out to optimize the parameters of the deformations in relation to a predefined script.

Acknowledgment

REFERENCES

[1] D. Yu and L. Deng, "Deep Convex Net: A scalable architecture for speech pattern classification," in *Proc. of INTERSPEECH*, 2011, pp. 2285–2288.

[2] D. Ciresan, U. Meier, L. M. Gambardella, and J. Schmidhuber, "Deep, big, simple neural nets for handwritten digit recognition," *Neural Computation*, vol. 12, pp. 3207–3220, 2010.

[3] G. E. Hinton and R. R. Salakhutdinov, "Reducing the dimensionality of data with neural networks," *Science*, vol. 313, pp. 504–507, July 2006.

[4] Y. LeCun, B. Boser, J. Denker, D. Henderson, R. Howard, W. Hubbard, and L. Jackel, "Handwritten digit recognition with a back-propagation network," in *Advances in Neural Information Processing Systems (NIPS) 2*, 1990, pp. 396–404.

[5] D. Ghosh, T. Dube, and A. Shivaprasad, "Script recognition: A review," *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 32, no. 12, pp. 2142–2161, Dec. 2010.

[6] R. Plamondon and N. S. Srihari, "On-line and off-line handwritten recognition: a comprehensive survey," *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 22, no. 1, pp. 63–84, Jan. 2000.

[7] D. Cireşan, U. Meier, and J. Schmidhuber, "Multi-column deep neural networks for image classification," in *Computer Vision and Pattern Recognition (CVPR)*, 2012, pp. 3642–3649.

[8] D. DeCoste and B. Schölkopf, "Training invariant support vector machines," *Machine Learning*, vol. 46, no. 1-3, pp. 161–190, 2002.

[9] F. Lauer, C. Y. Suen, and G. Bloch, "A trainable feature extractor for handwritten digit recognition," *Pattern Recognition*, vol. 40, no. 6, pp. 1816–1824, 2007.

[10] X.-X. Niu and C. Y. Suen, "A novel hybrid cnn-svm classifier for recognizing handwritten digits," *Pattern Recognition*, vol. 45, no. 6, pp. 1318–1325, 2012.

[11] N. Arica and F. T. Yarman-Vural, "An overview of character recognition focused on off-line handwriting," *IEEE Trans. Systems, Man, and Cybernetics, Part C: Applications and Reviews*, vol. 31, no. 2, pp. 216–233, May 2001.

[12] D. Keysers, T. Deselaers, C. Gollan, and H. Ney, "Deformation models for image recognition," *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 29, no. 8, pp. 1422–1435, Aug. 2007.

[13] T. Cover and P. Hart, "Nearest neighbor pattern classification," *IEEE Trans. Information Theory*, vol. 13, no. 1, pp. 21–27, 1967.

[14] M. Muja and D. G. Lowe, "Scalable nearest neighbor algorithms for high dimensional data," *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 36, no. 11, pp. 2227–2240, Nov. 2014.

[15] J. Dean and S. Ghemawat, "Mapreduce: simplified data processing on large clusters," *Communications of the ACM*, vol. 51, no. 1, pp. 107–113, Jan. 2008.

[16] W. Lu, Y. Shen, S. Chen, and B. C. Ooi, "Efficient processing of k nearest neighbor joins using MapReduce," in *Proc. of the VLDB Endowment VLDB Endowment Hompage archive*, vol. 5, June 2012, pp. 1016–1027.

[17] C. Zhang, F. Li, and J. Jestes, "Efficient parallel kNN joins for large data in mapreduce," in *Proc. of the 15th Int. Conf. on Extending Database Technology (EDBT)*, 2012, pp. 38–49.

[18] S. Uchida and H. Sakoe, "Survey of elastic matching techniques for handwritten character recognition," *IEICE Trans. Information and Systems*, vol. 88, no. 8, pp. 1781–1790, 2005.

[19] S. Gold, A. Rangarajan, C. P. Lu, S. Pappu, and E. Mjolsness, "New algorithms for 2-d and 3-d point matching: pose estimation and correspondence," *Pattern Recognition*, vol. 31, no. 8, pp. 1019–1031, 1998.

[20] J. Nagi, F. Ducatelle, G. A. Di Caro, D. Ciresan, U. Meier, A. Giusti, F. Nagi, J. Schmidhuber, and L. M. Gambardella, "Max-pooling convolutional neural networks for vision-based hand gesture recognition," in *Proc. of the IEEE Int. Conf. on Signal and Image Processing Applications*, 2011, pp. 342–347.

[21] J. Canny, "A computational approach to edge detection," *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 8, no. 6, pp. 679–698, Nov. 1986.

[22] T. K. Ho, J. J. Hull, and S. N. Srihari, "Decision combination in multiple classifier systems," *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 16, pp. 66–75, Jan. 1994.

[23] A. Rahman and M. Fairhurst, "Multiple classifier decision combination strategies for character recognition: A review," *Int. J. of Document Analysis and Recognition (IJDAR)*, vol. 5, no. 4, pp. 166–194, 2003.

[24] S. Tulyakov, S. Jaeger, V. Govindaraju, and D. Doermann, "Review of classifier combination methods," in *Machine Learning in Document Analysis and Recognition*, ser. Studies in Computational Intelligence, S. Marinai and H. Fujisawa, Eds.   Springer Berlin Heidelberg, 2008, vol. 90, pp. 361–386.

[25] H. Cecotti and A. Belaid, "Hybrid OCR combination approach complemented by a specialized ICR applied on ancient documents," in *Proc. of the 8th Int. Conf. on Document Analysis and Recognition (ICDAR)*, vol. 2, 2005, pp. 1045–1049.

[26] N. Otsu, "A threshold selection method from gray-level histograms," *IEEE Trans. Sys. Man. Cyber.*, vol. 9, no. 1, pp. 62–66, 1979.

[27] B. B. Chaudhuri and U. Pal, "A complete printed Bangla OCR system," *Pattern Recognition*, vol. 31, pp. 531–549, 1998.

[28] U. Pal and B. B. Chaudhuri, "Indian script character recognition: a survey," *Pattern Recognition*, vol. 37, no. 9, pp. 1887–1899, Sept. 2004.

[29] U. Bhattacharya and B. Chaudhuri, "Databases for research on recognition of handwritten characters of indian scripts," in *Proc. of the 8th Int. Conf. on Document Analysis and Recognition (ICDAR'05)*, 2005, pp. 789–793.

[30] B. Chaudhuri, "A complete handwritten numeral database of Bangla - a major Indic script," in *Proc. of the 10th Int. Workshop on Frontiers in Handwriting Recognition (IWFHR'10)*, 2006, pp. 1–6.

[31] S. Vajda, K. Roy, U. Pal, B. Chaudhuri, and A. Belaïd, "Automation of Indian postal documents written in Bangla and English," *Int. Journal of Pattern Recognition and Artificial Intelligence*, vol. 23, no. 8, pp. 1599–1632, Dec. 2009.

[32] I. K. Sethi and B. Chatterjee, "Machine recognition of constrained hand printed devanagari," *Pattern Recognition*, vol. 9, pp. 69–75, July 1977.

[33] T. Bhowmick, S. Parui, U. Bhattacharya, and B. Shaw, "An HMM based recognition scheme for handwritten oriya numerals," in *Proc. of the 9th Int. Conf. on Information Technology (ICIT 2006)*, 2006, pp. 105–110.

[34] U. Bhattacharya, M. Shridhar, S. Parui, P. Sen, and B. Chaudhuri, "Offline recognition of handwritten Bangla characters: an efficient two-stage approach," *Pattern Analysis and Applications*, vol. 15, no. 4, pp. 445–458, 2012.

[35] F. Kimura, K. Takashina, S. Tsuruoka, and Y. Miyake, "Modified quadratic discriminant functions and the application to Chinese character recognition," *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 9, pp. 149–153, 1987.

[36] S. Mandal, S. Sur, A. Dan, and P. Bhowmick, "Handwritten Bangla character recognition in machine-printed forms using gradient information and Haar wavelet," in *Proc. of the 2011 Int. Conf. on Image Information Processing (ICIIP)*, 2011, pp. 1–6.

[37] U. Bhattacharya, M. Shridhar, and S. Parui, "On recognition of handwritten bangla characters," in *Proc. of the 5th Indian Conf. on Computer Vision, Graphics and Image Processing (ICVGIP)*, 2006, pp. 817–828.

[38] H. Cecotti, M. P. Eckstein, and B. Giesbrecht, "Single-trial classification of event-related potentials in rapid serial visual presentation tasks using supervised spatial filtering," *IEEE Trans. Neural Networks and Learning Systems*, vol. 15, pp. 2030–42, Nov. 2014.

[1]http://www.isical.ac.in/~ujjwal/download/database.html