Engineering Applications of Artificial Intelligence 45 (2015) 405–414

Contents lists available at ScienceDirect

Engineering Applications of Artificial Intelligence

journal homepage: www.elsevier.com/locate/engappai

Recognition of handwritten characters using local gradient feature descriptors

Olarik Surinta *,[1], Mahir F. Karaaba [1], Lambert R.B. Schomaker [1], Marco A. Wiering [1]

Institute of Artificial Intelligence and Cognitive Engineering (ALICE), University of Groningen, PO Box 407, 9700 AK Groningen, The Netherlands

ARTICLE INFO

Article history:
Received 14 April 2015
Received in revised form
5 June 2015
Accepted 23 July 2015
Available online 10 August 2015

Keywords:
Handwritten character recognition
Feature extraction
Local gradient feature descriptor
Support vector machine
k-nearest neighbors

ABSTRACT

In this paper we propose to use local gradient feature descriptors, namely the scale invariant feature transform keypoint descriptor and the histogram of oriented gradients, for handwritten character recognition. The local gradient feature descriptors are used to extract feature vectors from the handwritten images, which are then presented to a machine learning algorithm to do the actual classification. As classifiers, the *k*-nearest neighbor and the support vector machine algorithms are used. We have evaluated these feature descriptors and classifiers on three different language scripts, namely Thai, Bangla, and Latin, consisting of both handwritten characters and digits. The results show that the local gradient feature descriptors significantly outperform directly using pixel intensities from the images. When the proposed feature descriptors are combined with the support vector machine, very high accuracies are obtained on the Thai handwritten datasets (character and digit), the Latin handwritten datasets (character and digit), and the Bangla handwritten digit dataset.

1. Introduction

Handwritten character recognition systems have several important applications, such as zip-code recognition, writer identification for e.g. forensic research, searching in historical manuscripts, and others. For such applications, the system should be able to recognize handwritten characters written on many different kinds of documents, such as contemporary or historical manuscripts. The aim is to let the system to automatically extract and recognize the characters that are embedded in the manuscript. The quality of the manuscript is one of the factors that can improve the recognition accuracy (Gupta et al., 2011). It is essential to deal with the different problems that occur in the manuscripts, such as distortions in a character image and the background noise that can appear during the scanning process. The aim of our work is to develop new algorithms that can obtain a high recognition accuracy.

Obtaining high recognition accuracies on handwritten character datasets is a challenging problem, for which many different solutions have been proposed. Although on the standard MNIST dataset extremely high accuracies have been obtained (Meier, 2011), there are many other datasets consisting of less examples and which can be considered more difficult. These datasets are challenging due to different writing styles of the same characters, different writing persons (with differences in age, gender, and education), different writing devices, and difficulties due to background noise that appears from the printer (Surinta et al., 2012).

In this paper we emphasize the importance of the recognition of complex handwritten Thai, Bangla, and Latin scripts, for which the handwritten characters and digits are highly varying due to different shapes, strokes, curls, and concavities (Mandal et al., 2011). Some samples of the handwritten characters are shown in Fig. 1. Note that the handwritten images shown in this paper are resized to the same resolution for illustration purposes. Due to the high variety, the direct use of pixel intensities may not work very well, because there is sometimes little overlap between two handwritten images displaying the same character. Therefore, in this paper we propose to use feature extraction techniques which are robust to local displacements, but still provide discriminative feature vectors as representation of the handwritten characters. The feature extraction methods that we will make use of have also been extensively used for object recognition, namely the scale invariant feature transform (SIFT) descriptor (Lowe, 2004) and the histogram of oriented gradients (HOG) (Dalal and Triggs, 2005). This paper shows that the use of these local gradient feature descriptors to extract features from handwritten characters and digits leads to a very well performing system. High recognition performances are obtained on the challenging handwritten datasets even with a simple classifier such as the *k*-nearest neighbor method, and very high recognition accuracies are obtained when using a support vector machine classifier.

* Corresponding author. Tel.: +31 50 363 9464.
E-mail addresses: o.surinta@rug.nl (O. Surinta), m.f.karaaba@rug.nl (M.F. Karaaba), l.r.b.schomaker@rug.nl (L.R.B. Schomaker), m.a.wiering@rug.nl (M.A. Wiering).
URL: http://www.ai.rug.nl/~mrolarik (O. Surinta).

[1] ALICE http://www.rug.nl/research/alice

http://dx.doi.org/10.1016/j.engappai.2015.07.017
0952-1976/© 2015 Elsevier Ltd. All rights reserved.

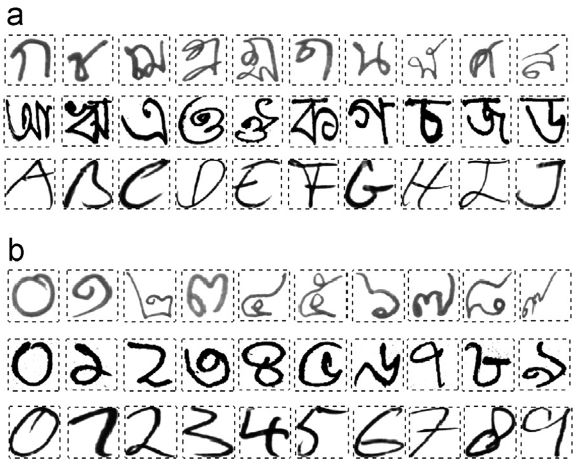

Fig. 1. Some examples of the Thai, Bangla, and Latin handwritten scripts as shown in the first, second, and third rows, respectively. Sample of (a) handwritten characters, and (b) handwritten digits.

Related work: In previous studies, the raw image (IMG) method, which directly copies the intensities of the pixels of the ink trace (Surinta et al., 2013), has often been used as the feature extraction method. It extracts a high dimensional feature vector that depends on the size of the input image.

In recent years, deep learning architectures (Hinton et al., 2006; Schmidhuber, 2015) have been effectively used for handwritten digit recognition. Most of the studies have focused on the benchmark MNIST dataset (LeCun and Cortes, 1998) and achieved high accuracies such as higher than 98% or 99%. The MNIST dataset consists of isolated handwritten digits with size of 28×28 pixel resolution and contains 60,000 training images and 10,000 test images. In Hinton et al. (2006), a greedy training algorithm is proposed for constructing a multilayer network architecture which relies on the restricted Boltzmann machine, called deep belief networks (DBN). The performance obtained from the DBN with three hidden layers (500–500–2000 hidden units) on the MNIST dataset was 98.75%. This accuracy is higher than obtained with a multi-layer perceptron and a support vector machine (SVM).

Furthermore, the convolutional neural network (CNN) (LeCun et al., 1998) is used as a feature extraction and classification technique, and the accuracy obtained is 99.47% (Jarrett et al., 2009). In another CNN-based method (Cireşan et al., 2011), the committee technique is proposed. Here multiple CNNs are combined in an ensemble, for which different CNNs are trained on different pixel resolutions of the images. The images in the dataset are rescaled from 28×28 ($N \times N$) to $N=10$, 12, 14, 16, 18, and 20 pixel resolutions. Then, 7-net committees are used. This method obtained the high accuracy of 99.73% on MNIST. However, a single CNN in their work is reported to take approximately 1–6 h for training on a graphics processing unit (GPU) and the 7-net committees are seven times slower than a single CNN. The best technique for the MNIST dataset uses an ensemble of 35-net committees (Cireşan et al., 2012). This technique obtained the very high accuracy of 99.77%. Although such high recognition performances are sometimes achieved, these methods require large training sets and long training times to make the recognition system work well.

For handwritten Bangla digit recognition, Liu and Suen (2009) proposed to use the local gradient directions of local strokes, called the gradient direction histogram feature. The feature vectors are extracted from an image and then given to a classifier. The recognition performance of the best classifier is 99.40% on the ISI Bangla numerals dataset (Chaudhuri, 2006) which contains 19,329 training images and 4000 test images. Compared to the MNIST dataset, ISI Bangla numerals dataset is more difficult due to background noise and more different types of handwriting.

In our research, we are interested in novel methods that obtain high recognition accuracies without the availability of many training examples, and which also do not require a huge amount of training time or high performance computing algorithms.

Contributions of our paper: This paper first of all provides a new standard Thai handwritten character dataset for comparison of feature extraction techniques and methods. In this paper we will make use of three complex datasets in total, namely Bangla, Thai, and Latin, for which very high recognition accuracies have not been obtained before. This is due to the difficult problems of the Thai and the Bangla handwritten scripts such as the complex structural characteristics of the characters, the similarities between the character sets (see Fig. 7(a) and (b)), the similar structures between different characters (see Fig. 4), and the background noise. These factors negatively affect the performance of a handwritten character recognition system.

To address the problems mentioned, two local gradient feature descriptors that extract feature vectors from the challenging handwritten character images are proposed, namely the scale invariant feature transform keypoint descriptor (Lowe, 2004) and the histogram of oriented gradients (Dalal and Triggs, 2005). The feature descriptors compute feature vectors with image filters such as the Sobel filter and the Gaussian filter. Subsequently, the orientations within each region are calculated and weighted into an orientation histogram. Because these feature descriptors are invariant to small local displacements, the descriptors provide robust feature vectors.

These feature extraction methods extract features, which are then used as input for a classifier. In this paper, we experimented with two different classifiers: a k-nearest neighbor classifier and a support vector machine, so that we can also compare performance differences between these machine learning methods. We evaluate the methods on the three handwritten character scripts: Thai, Bangla, and Latin, for which we use both the handwritten characters and the handwritten digits. To show the importance of using the proposed local gradient feature descriptors, we have compared them to a method that directly uses pixel intensities of the handwritten images (called the IMG method). The results show that the feature descriptors with the support vector machine obtain very high recognition performances on the datasets, whereas the use of the IMG method performs much worse.

Paper outline: This paper is organized in the following way. Section 2 describes the local gradient feature descriptors. Section 3 describes the two classifiers including the k-nearest neighbors algorithm as a simple classifier and the support vector machine algorithm with the radial basis function kernel as a more powerful classifier. The handwritten character datasets which are used in the experiments, namely Thai, Bangla, and Latin scripts, are described in Section 4. The experimental results of the different combinations of feature descriptors and classifiers are presented in Section 5. The conclusion and some directions for future work are given in the last section.

2. Local gradient feature descriptors

To study the effectiveness of local gradient feature descriptors for handwritten character recognition, we compare two existing feature extraction techniques, namely the histogram of oriented gradients and the scale invariant feature transform keypoint descriptor. Moreover, these local gradient feature descriptors are compared to the IMG method. The IMG method uses the raw pixel intensities of the handwritten images and is a simple and widely used method. In this study, the handwritten images are resized to two pixel resolutions, 28×28 and 36×36, so that for the IMG method 784 and 1296 feature values are computed, respectively.

2.1. Histograms of oriented gradients (HOG)

The HOG descriptor was first introduced in Dalal and Triggs (2005) for detecting a human body in an image. It has become very successful in diverse domains such as face, pedestrian, and on-road vehicle detection (Déniz et al., 2011; Lee et al., 2013; Arróspide et al., 2013). The HOG descriptor is originally defined as the distribution of the local intensity gradients from an image, which are computed from small connected regions (*cells*). We will now present the details of the HOG image descriptor.

The HOG feature vector is computed from the image using gradient detectors. In this paper, each pixel is convolved with the simple convolution kernel as follows:

$$G_x = f(x+1,y) - f(x-1,y)$$

$$G_y = f(x,y+1) - f(x,y-1) \tag{1}$$

G_x and G_y are the horizontal and vertical components of the gradients, respectively. In our experiments, the HOG descriptor is calculated over rectangular blocks (R-HOG) with non-overlapping blocks.

To ignore negative gradient directions, the range of gradient orientations is defined between 0° and 180° (Dalal and Triggs, 2005; Arróspide et al., 2013). The gradient magnitude M and the gradient orientation θ are calculated by

$$M(x,y) = \sqrt{G_x^2 + G_y^2}$$

$$\theta(x,y) = \tan^{-1}\frac{G_y}{G_x} \tag{2}$$

After this, histograms are computed from the occurrences of oriented gradients across large structures (*blocks*) of the image as shown in Fig. 2. The gradient orientations are stored into 9 orientation bins β.

The combination of the histograms from each block represents the feature descriptor. The feature vector size of the HOG descriptor depends on the selected numbers of blocks and bins. It has been shown that the performance of the HOG descriptor depends mostly on the number of blocks (Déniz et al., 2011).

Finally, the feature descriptors are normalized by applying the L2 block normalization (Lee et al., 2013) as follows:

$$V'_k = \frac{V_k}{\sqrt{\|V_k\|^2 + \varepsilon}} \tag{3}$$

where V_k is the combined histogram from all block regions, ε is a small value close to zero, and V'_k is the normalized HOG descriptor feature vector.

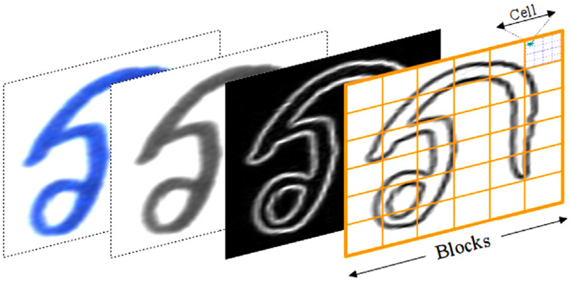

Fig. 2. Example of the rectangular HOG descriptor. The third image shows the gradient magnitude image after applying the simple convolution kernel to the second image. The fourth image shows the partitioning of the image into 6 × 6 non-overlapping blocks. Here, each block provides a separate angular histogram for the gradient orientations, which are afterwards concatenated and normalized.

2.2. Scale invariant feature transform descriptor (siftD)

The scale invariant feature transform (SIFT) descriptor was described in Lowe (2004) and is quite similar to the HOG descriptor, but there are some important differences as well, which we will explain later. The siftD descriptor computes 128 dimensional feature vectors for each keypoint (Sun et al., 2014). The detected keypoints in the standard SIFT algorithm are computed so that they are invariant to different translations, scales, rotations and they are also robust to other local geometric distortions. Additionally, for each keypoint a translation, scale, and orientation value are computed. The SIFT method is widely used in object, scene, and face recognition (Abdullah et al., 2009; Seo and Park, 2014).

To extract feature vectors from images, the standard SIFT algorithm detects keypoints that correspond to the local extrema of the Difference of Gaussians (DoG) function applied to the image with different scales. The problem of the standard SIFT algorithm is that in processing a character image, the number of detected keypoints will be variable. Therefore, the feature vectors are of variable size and different methods to handle this such as visual codebooks need to be used. However, the character images are well defined and well segmented. Therefore, in this study, we will only use the 128-dimensional descriptor, at *given* locations, e.g. the center of a character box (see Fig. 3). In order to determine whether this provides a sufficient resolution, additional experiments with more predefined keypoint centers will be performed, yielding higher-dimensional siftD feature vectors.

The siftD descriptor performs the following steps to extract the features from a handwritten character image. First, the input image is smoothed by a convolution with a variable-scale Gaussian kernel:

$$L(x,y,\sigma) = G(x,y,\sigma) * I(x,y) \tag{4}$$

where $I(x,y)$ is the pixel intensity at location x, y in the input image and $G(x,y,\sigma)$ is the Gaussian kernel. The parameter σ determines the width of the Gaussian kernel and is set to 0.8 in our experiments. Then, the horizontal and vertical components of the gradients G_x and G_y are computed according to Eq. (5). Afterwards, the magnitude $M(x,y)$ and orientation $\theta(x,y)$ for each Gaussian smoothed image pixel are computed according to Eq. (2).

$$G_x = L(x+1,y,\sigma) - L(x-1,y,\sigma)$$
$$G_y = L(x,y+1,\sigma) - L(x,y-1,\sigma) \tag{5}$$

The main image is split into 4 × 4 subregions (blocks) and then for each block an orientation histogram is made. The orientation histogram uses 8 bins which cover 360°, which results in 128 dimensions for the feature vector if one main region (consisting of 4 × 4 subregions) is used. The orientation histogram is weighted by gradient magnitudes and a Gaussian-weighted circular window (Lowe, 2004).

It should be noted that the proposed use of siftD is somewhat similar to the HOG method, which is also orientation based. However, there are still a number of differences (1) the HOG descriptor uses

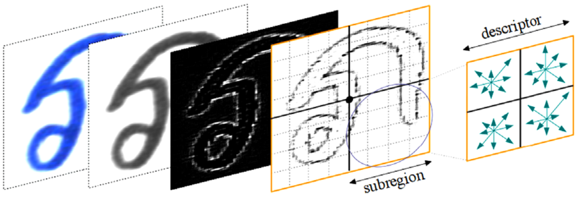

Fig. 3. Overview of the SIFT descriptor. The illustration shows a character and one main region. The region is divided into 4 × 4 subregions (only 2 × 2 are shown). One subregion subsequently provides a descriptor which is represented as 8 orientation bins as shown on the right, yielding a 128-dimensional feature vector (dubbed siftD, here).

absolute angles between 0 and 180° and siftD uses all angles between 0 and 360°, (2) in HOG all pixels are weighted equally in the rectangular blocks whereas in the siftD descriptor the influence of local gradients of different pixels is computed by weighting the distance of the pixel to the keypoint (center of the region), and (3) siftD uses a Gaussian filter before extracting the gradient orientations and magnitudes.

3. Classification algorithms

In the following we provide a description of the classifier methods that are used in the experiments, namely the *k*-nearest neighbor classifier and the support vector machine.

3.1. k-nearest neighbors algorithm (kNN)

*k*NN is classified as an instance based learning algorithm, which is suitable for large amounts of data. It is a well known non-parametric and simple algorithm. The *k*NN algorithm has been used in statistical estimation, scene recognition (Abdullah et al., 2010) and also writer identification systems (Brink et al., 2012). In some previous studies, the *k*NN algorithm has been used in character recognition and a good recognition performance was obtained. In Kumar et al. (2011), the authors proposed two feature extraction techniques including diagonal and transition feature extraction, and then experimented on the Gurmakhi dataset. The recognition performance with this method is 94.12%. In Rakesh Rathi et al. (2012), the authors used feature mining algorithms to compute the feature vector and tested the method on the Devanagari vowels database. Here, the KNN algorithm is used as a classification technique, and obtained the accuracy of 96.14%.

In *k*NN, the input vector is compared with training samples to compute the most similar *k* neighbors. The efficacy of the *k*NN algorithm depends on two key factors: a suitable distance function and the value of the parameter *k*. In this study, the Euclidean distance is selected as the function in order to calculate distance values from an input vector *x* to each training sample *y*. The Euclidean distance is calculated by the following equation:

$$d(x,y)=\sqrt{\sum_{i=1}^{N}(x^i-y^i)^2} \quad (6)$$

where *N* is the number of dimensions of *x* and *y*. Then distances between the input vector and the training samples are compared to identify the closest neighbors to the input vector. The parameter *k* is usually chosen as an odd number, e.g. if parameter $k=3$, the three closest neighbors are considered in order to determine the class for a particular input vector. Let $Z=\{(y_i,c_i)\}$ be the set of *M* labelled training samples, where $y_i\in\mathbf{R}^N$ and $c_i\in C$ and *C* is the set of class labels present in the training samples. In the classification stage for an unknown sample *x*, first the distance $d(x,y_i)$ from *x* to each sample in *Z* is calculated according to Eq. (6). Let $D_k=\{d_1,d_2,\ldots,d_k\}$ be the set of *k* nearest distances for the input *x*, where $d_1\le d_2\le\cdots\le d_k$. To classify an unknown sample *x*, the number of occurrences each class belongs to the input vectors in D_k is counted, and finally the most frequently occurring class is selected as the output of the classifier (for which ties are broken randomly).

3.2. Support vector machine (SVM)

The support vector machine (SVM) algorithm invented by Vapnik (1998) has been effectively applied to many pattern recognition problems. The algorithm finds the optimal separating hyperplane, which has the maximum distance to the training points that are closest to the hyperplane. The training points closest to the computed separating hyperplane are called support vectors. The original SVM is a linear binary classifier, which is useful for two-class classification problems. On the other hand, it does not provide good separation for non-sparse complex data (e.g. image data). We will now shortly describe the workings of the SVM. Let *D* be a training dataset

$$D=\{(x_i,y_i),1\le i\le M\} \quad (7)$$

where $x_i\in\mathbf{R}^N$ are input vectors and $y_i\in\{+1,-1\}$ is the binary label of pattern x_i. The optimal model from the set of hyperplanes in $\mathbf{R}^N$ is computed by the SVM optimization algorithm. The decision function is given by

$$f(x)=\mathrm{sign}(w^Tx+b) \quad (8)$$

where *w* is the weight vector orthogonal to the hyperplane and *b* is the bias value. To compute the parameters *w* and *b*, the SVM algorithm minimizes the following cost function:

$$J(w,\xi)=\frac{1}{2}w^Tw+C\sum_{i=1}^{n}\xi_i \quad (9)$$

subject to constraints

$$w^Tx_i+b\ge+1-\xi_i \quad \text{for } y_i=+1$$

and

$$w^Tx_i+b\le-1+\xi_i \quad \text{for } y_i=-1$$

where *C* controls a trade-off between training error and generalization, and $\xi_i\ge0$ are slack variables which tolerate some errors, but which need to be minimized. While this soft margin method is useful to fit a model to a complex dataset, if used improperly, overfitting can occur.

The maximum margin splits the hyperplane with $w^Tx+b=0$. The splitting hyperplane obtains the largest distance to the closest positives $w^Tx+b=+1$ and negatives $w^Tx+b=-1$. The linear kernel function is defined as follows:

$$K(x_i,x_j)=x_i^Tx_j \quad (10)$$

The linear SVM algorithm has been extended to deal with non-linear classification problems. Many non-linear kernel functions have been proposed. In this paper we choose the radial basis function (RBF) kernel as a non-linear similarity function in the SVM classifier. The RBF kernel computes the following similarity value between two input vectors:

$$K(x_i,x_j)=\exp(-\gamma\|x_i-x_j\|^2) \quad (11)$$

where γ is a kernel parameter of the RBF kernel. Large values of the γ parameter may cause overfitting due to the increase of the number of support vectors.

For multi-class problems, we use the one-vs-all strategy. In this method, the input vector is given to all SVM models which are constructed for each class. Then the class with the maximum discriminant output is selected from these models as the winning class. The idea here is that every model is constructed to discriminate between a class and the other classes (Liu et al., 2003).

4. Handwritten character datasets

We evaluate the different handwritten character recognition methods on three isolated handwritten script datasets belonging to three languages (Thai, Bangla, and Latin), all of which are composed of handwritten characters and digits. The original handwritten scripts in the datasets are not normalized to a fixed-size image and therefore are in numerous pixel space resolutions. Furthermore, we have manually collected a new Thai handwritten script dataset that contains 24,045 character images in total from various writers. The details of the Thai handwritten dataset are described in Section 4.1.

In order to prepare the handwritten character images, a few simple preprocessing steps are applied. The handwritten images are first converted from the color image format into a gray-scale image. Subsequently, the sample images are normalized into 28×28 and 36×36 pixel resolution with aspect ratios preserved. The experiments on the different pixel resolutions of the handwritten images are described in Section 5.

4.1. Thai handwritten dataset

The number of Thai consonants is not uniquely defined, because some characters are outdated. In this research, the Thai handwritten dataset is collected according to the standard Thai script consisting of 78 characters. On the other hand, Nopsuwanchai et al. (2006) presented a ThaiCAM database that contains a different Thai script of 87 characters. This script includes some extra obsolete characters and special symbols, which are not essential for writing. Generally, the writing style of several Thai characters is very similar, but there are some differences in some details such as head, loop, curl, and concavity as shown in Fig. 4. Some character recognition systems use local features to extract information of the characters (Phokharatkul et al., 2007). However, some important details can disappear because of the writing styles. Various writing styles of Thai handwritten characters are illustrated in Fig. 5.

The performances obtained with previous approaches have not reached very high recognition rates. Nopsuwanchai et al. (2006) proposed block-based principal component analysis and composite images and used a hidden Markov model (HMM) as a classification technique. They obtained 92.03% accuracy on the ThaiCAM database. Some hybrid techniques of heuristic rules and neural networks are employed in Mitrpanont and Imprasert (2011). The performance obtained from this approach on the Thai handwritten character dataset was 92.78%.

We collected a new Thai handwritten script dataset from 150 native writers who studied in the university and are aged from 20 to 23 years old. They used a 0.7 mm ink pen writing Thai scripts consisting of consonants, vowels, tones and symbols on a prepared A4 form. The participants were allowed to write only the isolated Thai script on the form and at least 100 samples per character. We allowed writers to write in various styles without pressure. However, the character images obtained from this dataset generally have no background noise. Moreover, the forms were scanned at a resolution of 200 dots per inch and stored as color images. Finally, we have used an uncomplicated line and character segmentation method based on the horizontal and vertical projection profile to separate and crop the isolated characters. The Thai handwritten dataset is available from http://www.ai.rug.nl/~mrolarik/THI/ for research purposes.

Thai handwritten character dataset (*THI-C*68): This dataset consists of 13,130 training samples and 1360 test samples randomly selected from the main dataset. Sample images of the dataset are illustrated in Fig. 6(a). In this research, we have selected the standard Thai script of 68 Thai characters, which are composed of 44 consonants, 17 vowels, 4 tones and 3 symbols.

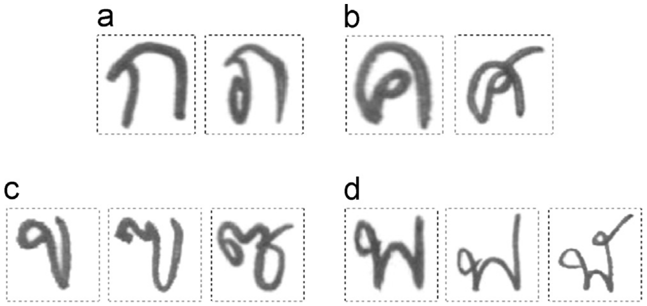

Fig. 4. Illustration of the relation between different Thai characters. In (a) and (b), the second character is constructed by slightly changing the first (different) character. In (c) and (d) the third is created by a modification of the second character, which is a modification of the first character.

Thai handwritten digit dataset (*THI-D*10): This dataset has 9555 samples including 8055 training samples and 1500 test samples. The total number of samples in each class is larger than 900. Sample characters of this dataset are shown in Fig. 6(b).

4.2. Bangla handwritten dataset

Bangla (or Bengali) is the second most popular language in India and Bangladesh and the fifth most used language around the globe (Pal et al., 2007). The Bangla basic script consists of 11 vowels and 39 consonants (Bhowmik et al., 2009; Das et al., 2010). This paper deals with recognition of handwritten characters of 45 classes and handwritten digits of 10 classes from different writers. The Bangla handwritten dataset (Bhowmik et al., 2009) in this study has a large diversity of writing styles and some characters are nearly identical with other characters. The dataset contains different kinds of backgrounds, some of which are clear, but most are quite noisy. Finally, it contains a variety of pixel space resolutions. Hence, it is much more challenging than the well-known MNIST dataset (LeCun and Cortes, 1998). Fig. 7(a) and (b) shows the similarities between two different handwritten digits.

Bangla handwritten character dataset (*BANG-C*45): The Bangla basic character set includes 45 classes and contains 4627 character images in the training set and 900 examples in the test set. In this dataset the number of character images per class is around 100. Additionally, the characters in the dataset are in gray-scale format, and some of them have a noisy background. Some samples of the Bangla handwritten character dataset are shown in Fig. 8(a).

Bangla handwritten digit dataset (*BANG-D*10): The set of Bangla digits consists of 9161 instances in the training set and 1500 instances in the test set. We randomly selected 150 character images per class as a test set. Some examples of this dataset are shown in Fig. 8(b).

4.3. Latin handwritten dataset

The benchmark dataset for Latin handwritten character recognition is provided by van der Maaten (2009). The original images were collected by Schomaker and Vuurpijl (2000) for forensic writer identification and were named the *Firemaker* dataset. The handwritten text was written in Dutch script by 251 writers. It has 40,133 handwritten images and consists of uppercase characters and digits. In this dataset the capital letters are collected except for the 'X' letter. The Latin handwritten characters, called LATIN-*C*25, consist of 26,329 training samples and 11,287 test samples. The set of digits (LATIN-*D*10), which has less character images, consists of 1637 training samples and 880 test samples. Sample images of the Latin handwritten dataset are illustrated in Fig. 9.

An overview of the handwritten datasets is given in Table 1. The training set is used for 10-fold cross validation, splitting it according to the 9/1 rule. The test set is an independent hold-out set for an additional final evaluation.

5. Experimental results

We have compared the IMG method which directly uses pixel intensities to the two local gradient feature descriptors, namely the HOG descriptor and the siftD. The datasets are composed of isolated handwritten characters and digits. The handwritten images are converted to gray-scale and normalized to a fixed-size image. There

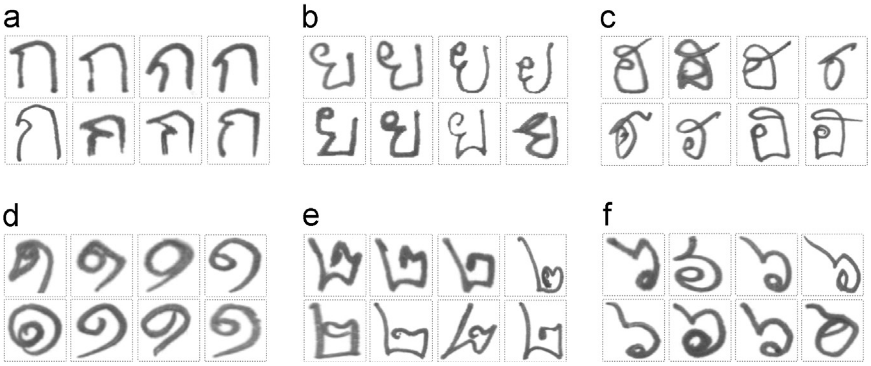

Fig. 5. Illustration of the diversity in the writing styles of the Thai handwritten dataset. (a), (b) and (c) show samples of Thai handwritten characters and (d), (e) and (f) show samples of Thai handwritten digits.

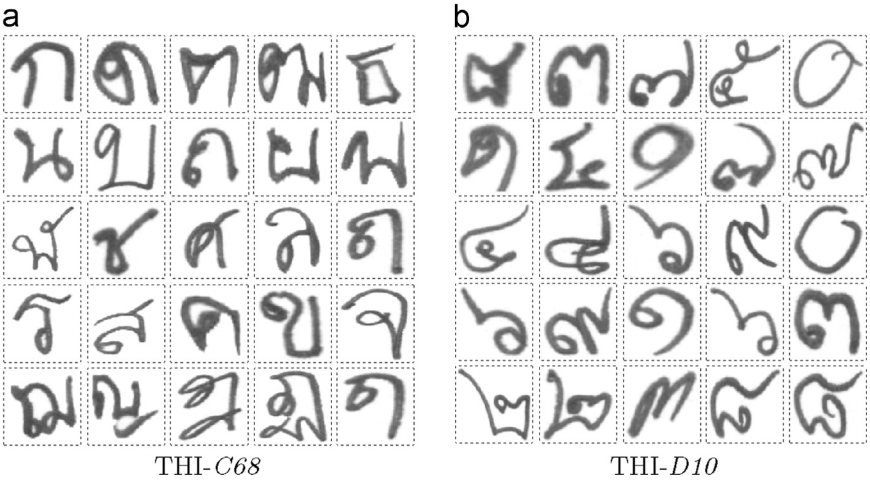

Fig. 6. Illustration of the Thai handwritten images. (a) Thai handwritten characters, and (b) Thai handwritten digits.

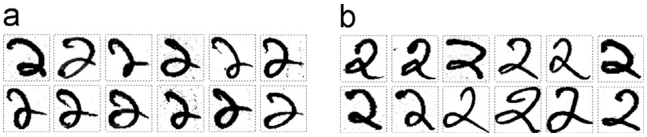

Fig. 7. Illustration of the similarities between different Bangla handwritten digits and the noisy background which appears in the Bangla handwritten dataset. (a), (b) Similarities of Bangla digits between number 1 and number 2, respectively.

are two pixel space resolutions which are used in these experiments: 28 × 28 and 36 × 36 pixel spaces.

In these experiments, we are using the recognition rate (accuracy) as our evaluation metric to measure the performance of each feature descriptor. For the experiments with *k*NN, the parameter *k* is selected from 1, 3, 5, and 7. For the SVM algorithm, grid-search with a logarithmic scale (Ben-Hur and Weston, 2010) is used to explore the two-dimensional parameter space of the SVM with the RBF kernel. Through grid-search the best combination of two parameters, C and γ, is then selected to create the model of the handwritten recognition systems. We use K-fold cross validation (cv) over the training set to prevent overfitting due to large γ and C parameter values. In this study, we use K-fold cross validation with $K=10$ for both classifiers.

5.1. Experiments with the HOG descriptor

We evaluated the performance of the HOG descriptor using several parameters. The parameters of the HOG descriptor include the block size $b_1 \times b_2$, the cell size $\eta_1 \times \eta_2$ and the number of orientation bins. The cell size parameters are defined as a square ($\eta_1 = \eta_2$). In our experiments we evaluate the use of 9 and 18 orientation bins. In our results, the orientation histogram with 9 bins slightly outperforms 18 bins. Furthermore, several block sizes are evaluated including $b=3$, 4, 6, 7, and 9, respectively.

The experimental results of the HOG descriptor on the handwritten datasets using the different numbers of feature dimensions ($N=81$, 144, 324, 441, and 729) are shown in Fig. 10. We evaluated the HOG descriptor on three handwritten character datasets including Thai (THI-*C68* and THI-*D10*), Bangla (BANG-*C45* and BANG-*D10*) and Latin (LATIN-*C25* and LATIN-*D10*). Fig. 10(a) shows the performance of the HOG descriptor using *k*NN with $k=5$, and Fig. 10(b) using the SVM with the RBF kernel for which standard values are used in this experiment ($C=1$ and $\gamma = 1/N$).

The results show that the HOG descriptor provides the highest recognition accuracies when the feature vector uses 324 dimensions. The performance is decreased slightly when the feature dimension is increased. In the following experiments, the HOG descriptor uses $N=324$ features, given by blocks of size 6 × 6 in the handwritten character images of size 36 × 36 with orientation histograms consisting of 9 bins ($6 \times 6 \times 9 = 324$).

5.2. Experiments with the SIFT keypoint descriptor (siftD)

For the SIFT keypoint descriptor experiments we evaluated the parameters of siftD. These are the pixel space, the number of keypoints and the region size. As mentioned before, in order to simplify the keypoint detection process, we divided the handwritten character image into small blocks ($b \times b$ blocks) and defined each center of a block as the keypoint. In these experiments, the number of keypoints

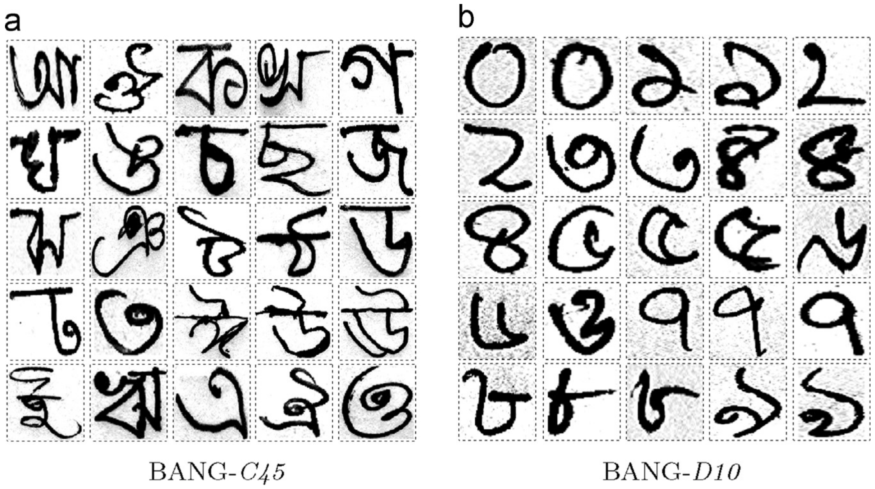

Fig. 8. Some examples of the Bangla handwritten dataset. (a) Bangla handwritten characters, and (b) Bangla handwritten digits.

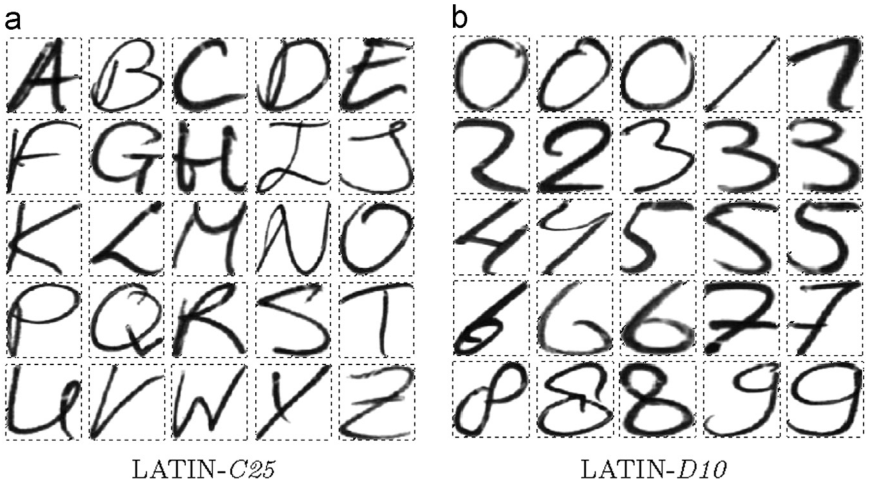

Fig. 9. Some examples of the Latin handwritten dataset. (a) Latin handwritten characters, and (b) Latin handwritten digits.

Table 1
Overview of the handwritten datasets.

Dataset	No. of classes	Train	Test
THI-*C*68	68	13,130	1360
THI-*D*10	10	8055	1500
BANG-*C*45	45	4627	900
BANG-*D*10	10	9161	1500
LATIN-*C*25	25	26,329	11,287
LATIN-*D*10	10	1637	880

is an important factor for obtaining the highest recognition accuracy. We tried out several keypoint numbers: 1 (1 × 1), 4 (2 × 2), 9 (3 × 3), and 16 (4 × 4). The feature dimensionality is associated with the number of keypoints.

It is important to emphasize that a high dimensionality of the input vector can decrease the recognition performance. Furthermore, the high dimensionality can make the system slow and causes a lot of memory usage during the training process. The results are overall best with 1 keypoint, the only better result obtained with 4 keypoints of siftD ($128 \times 4 = 512$ features) is with the *k*NN classifier for the Bangla character dataset as shown in Fig. 11(a) and (b). It is quite remarkable that the *k*NN and the SVM using siftD with one keypoint (128 features) perform so well as shown in Fig. 11.

5.3. Comparison of HOG and siftD to pixel intensities

We compared the IMG method to the local gradient feature descriptors on the challenging handwritten script datasets by using the *k*NN and the SVM as classifiers. The best feature descriptor parameter values from the previous experiments are selected. In contrast to the experiments before, here we have optimized the hyper-parameters of the classifiers. The best parameters found for these experiments are shown in Table 2.

The accuracy results of *k*NN are shown in Table 3. The *k*NN algorithm is selected in this paper, because we found it interesting to observe the performances obtained with a robust feature descriptor and a simple classifier. The performance of the *k*NN reaches above 95%, except on BANG-*C*45 using both feature descriptors. The experiments show that while the HOG descriptor performs better on the Latin handwritten dataset with the *k*NN, siftD is more powerful than the HOG descriptor on the other datasets (see Table 3). Most important, however, is that the results obtained with the proposed local gradient feature descriptors are much better than those obtained with the direct use of pixel intensities.

We also compared these results with the *k*NN classifier to previous results obtained on the handwritten Bangla digit dataset. In Surinta et al. (2013), the authors presented the unweighted majority voting method (UMV), which combines different SVM classifiers with the

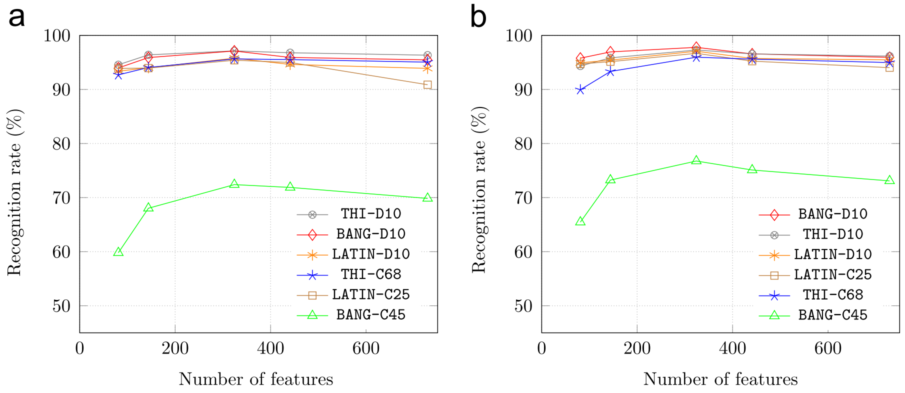

Fig. 10. A comparison of the performance (%) of the HOG descriptor on the handwritten datasets using different numbers of features. The experiments use (a) *k*NN with $k=5$ and (b) SVM with the RBF kernel. Here the RBF kernel parameters of the SVM algorithm are defined as $C=1$ and $\gamma = 1/N$.

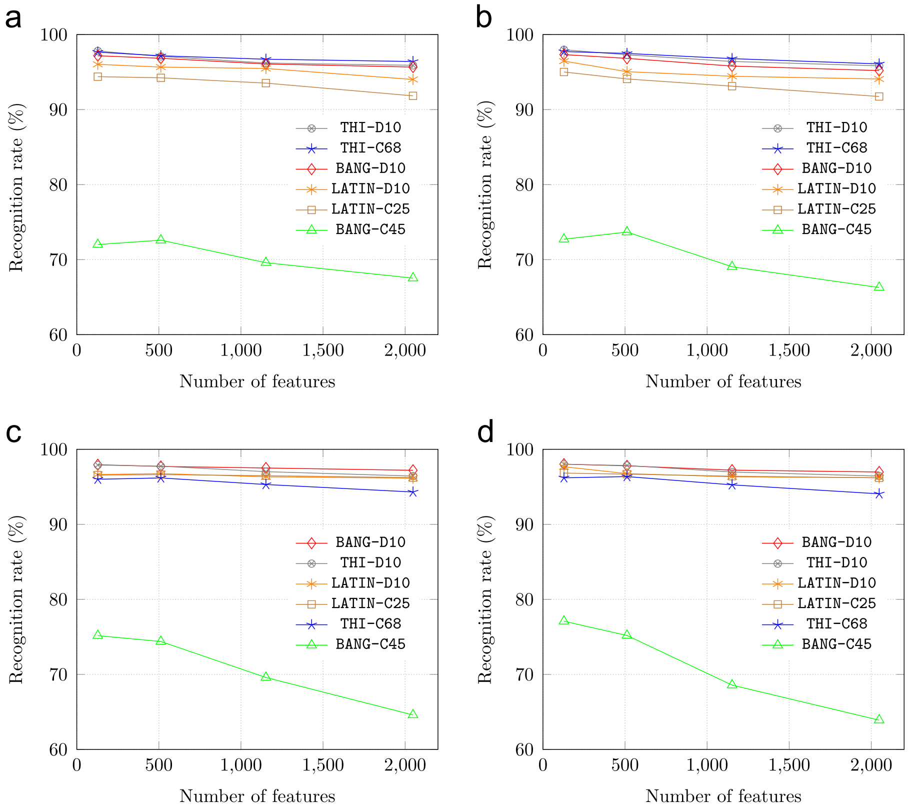

Fig. 11. A comparison of the performance (%) of siftD on the handwritten character datasets using different numbers of features and pixel resolutions. The pixel resolutions of the handwritten images are (left column) 28 × 28 and (right column) 36 × 36 pixels. The experiments (a), (b) use *k*NN with $k=3$ and (c), (d) SVM with the RBF kernel. The RBF kernel parameters of the SVM algorithm are defined as $C=1$ and $\gamma = 1/N$.

Table 2
The best training parameter values for the SVM with the RBF kernel and for the *k*NN algorithm.

Datasets	IMG		HOG		siftD		IMG	HOG	siftD
	SVM with RBF kernel						*k*NN		
	C	γ	C	γ	C	γ	k		
THI-C68	2^2	2^{-9}	2^2	2^{-6}	2^2	2^{-5}	1	3	3
THI-D10	2^2	2^{-9}	2^2	2^{-6}	2^2	2^{-5}	3	3	3
BANG-C45	2^2	2^{-10}	2^2	2^{-6}	2^4	2^{-7}	5	5	5
BANG-D10	2^1	2^{-9}	2^1	2^{-6}	2^2	2^{-5}	3	5	5
LATIN-C25	2^0	2^{-9}	2^2	2^{-7}	2^4	2^{-5}	1	5	3
LATIN-D10	2^0	2^{-9}	2^3	2^{-7}	2^4	2^{-6}	5	5	3

RBF kernel trained on different extracted features. This more complex ensemble method obtained 96.8%. Our current results show that the HOG descriptor and siftD obtain 97.11% and 97.35%, respectively. Because the HOG descriptor and siftD with the *k*NN method provide higher accuracies than the more complex method used in Surinta et al. (2013), these results demonstrate the effectiveness of the proposed local gradient feature descriptors.

We show the obtained results with the SVM classifier with the RBF kernel on the handwritten character datasets in Table 4. It can be seen from Table 4 that siftD is the best feature descriptor in our experiments on the three handwritten character datasets. The SVM with the RBF kernel outperforms the *k*NN with around 1–11% accuracy improvement. On the BANG-C45 dataset, the SVM with the RBF kernel increased the recognition performance with about 11% compared to the *k*NN classifier.

Table 3
The accuracy (%) and the standard deviation of the *k*NN classifier obtained with cross validation and on separate test sets. The results are computed using three handwritten character datasets.

Datasets	IMG (%)		HOG (%)		siftD (%)	
	10-cv	Test	10-cv	Test	10-cv	Test
THI-*C*68	93.55 ± 0.46	82.87	95.83 ± 0.76	88.31	**97.73** ± 0.44	91.91
THI-*D*10	93.52 ± 0.76	86.47	97.23 ± 0.51	93.73	**97.97** ± 0.50	97.83
BANG-*C*45	53.17 ± 1.96	46.11	72.40 ± 1.90	69.00	**74.50** ± 1.71	69.67
BANG-*D*10	91.05 ± 0.53	89.87	97.11 ± 0.44	95.60	**97.35** ± 0.74	96.07
LATIN-*C*25	88.00 ± 0.96	90.54	**95.40** ± 0.51	95.17	95.01 ± 0.62	96.12
LATIN-*D*10	91.76 ± 1.82	96.25	**97.79** ± 1.20	95.11	96.46 ± 1.25	96.48

Table 4
The SVM accuracy (%) and the standard deviation of handwritten character recognition experiments on the handwritten character datasets.

Datasets	IMG (%)		HOG (%)		siftD (%)	
	10-cv	Test	10-cv	Test	10-cv	Test
THI-*C*68	95.33 ± 0.08	90.59	98.42 ± 0.03	94.34	**98.93** ± 0.03	94.34
THI-*D*10	94.88 ± 0.09	88.53	98.58 ± 0.05	97.20	**99.07** ± 0.03	97.87
BANG-*C*45	63.25 ± 0.28	60.00	84.01 ± 0.33	82.78	**85.60** ± 0.18	85.00
BANG-*D*10	95.10 ± 0.09	94.87	98.73 ± 0.03	98.07	**98.91** ± 0.03	98.53
LATIN-*C*25	96.28 ± 0.03	95.94	97.79 ± 0.04	98.25	**98.23** ± 0.04	98.32
LATIN-*D*10	98.04 ± 0.12	96.36	98.10 ± 0.17	97.73	**98.58** ± 0.09	98.30

To summarize the results, the siftD and HOG feature descriptors strongly outperform the direct use of pixel intensities. Even with much less features (siftD computes 128 features for all datasets except for BANG-*C*45), very good results are obtained. Furthermore, the results demonstrate that the SVM significantly outperforms the *k*NN classifier. Finally, we want to mention that the SVM with the siftD method obtains very high recognition accuracies. On most datasets, cross validation accuracies around 99% are obtained. However, on the Bangla character dataset the performance of the HOG descriptor and siftD is much lower compared to the other datasets. This might be because of the image quality, the huge diversity in writing styles, the similarities between different characters, arbitrary used tail strokes which makes the definition of the bounding box around the characters harder, a high cursivity, and an insufficient number of handwritten training samples.

6. Conclusion

In this paper, we have demonstrated the effectiveness of local gradient feature descriptors for handwritten character recognition. The local gradient feature descriptors which we selected are siftD and HOG that extract the orientation histograms from the handwritten character gray-scale images. Only simple preprocessing schemes such as rescaling the image by preserving the aspect ratio and converting it from color to gray-scale were applied. We evaluated two machine learning techniques with the feature description methods on three different handwritten character datasets: Thai, Bangla, and Latin. The results show that the local gradient feature descriptors that convert the handwritten images to feature vectors are strong feature descriptors for handwritten character recognition problems. The siftD and the HOG descriptor give the best performances and significantly outperform the IMG method that directly uses pixel intensities (see Tables 3 and 4). Interestingly, siftD with only one keypoint (128 feature dimensions) outperforms the HOG descriptor (324 feature dimensions). The *k*NN and the SVM classifier have been compared, and the results show that the SVM significantly outperforms the *k*NN classifier.

Our results are better than the results reported in previous work, although it is sometimes hard to compare, because previous work has not experimented with all three datasets we used in this paper. In one related paper (Nopsuwanchai et al., 2006), their method obtained 92.03% on the ThaiCAM database. In another paper (Mitrpanont and Imprasert, 2011), their method obtained 92.78% on the Thai handwritten character dataset. Our best method obtains 94.34% on the test set and 98.93% with cross validation on the THI-*C*68 dataset and 97.87% on the test set and 99.07% with cross validation on the THI-*D*10 dataset. So our best method outperforms methods from previous works on the Thai handwritten character dataset.

In future work, we will concentrate on improving the handwritten character recognition performance on the Bangla character dataset. For this, we will study other feature descriptors which could even be more efficient and robust to a variation of the writing styles in this dataset and which can handle a small number of training examples.

References

Abdullah, A., Veltkamp, R.C., Wiering, M.A., 2009. Spatial pyramids and two-layer stacking SVM classifiers for image categorization: a comparative study. In: International Joint Conference on Neural Networks (IJCNN), pp. 5–12.

Abdullah, A., Veltkamp, R.C., Wiering, M.A., 2010. Fixed partitioning and salient points with MPEG-7 cluster correlograms for image categorization. Pattern Recognit. 43 (3), 650–662.

Arróspide, J., Salgado, L., Camplani, M., 2013. Image-based on-road vehicle detection using cost-effective histograms of oriented gradients. J. Vis. Commun. Image Represent. 24 (7), 1182–1190.

Ben-Hur, A., Weston, J., 2010. A user's guide to support vector machines. In: Carugo, O., Eisenhaber, F. (Eds.), Data Mining Techniques for the Life Sciences. Methods in Molecular Biology, vol. 609, Humana Press, pp. 223–239.

Bhowmik, T.K., Ghanty, P., Roy, A., Parui, S., 2009. SVM-based hierarchical architectures for handwritten Bangla character recognition. Int. J. Doc. Anal. Recognit. (IJDAR) 12 (2), 97–108.

Brink, A.A., Smit, J., Bulacu, M.L., Schomaker, L.R.B., 2012. Writer identification using directional ink-trace width measurements. Pattern Recognit. 45 (1), 162–171.

Chaudhuri, B.B., 2006. A complete handwritten numeral database of Bangla a major Indic script. In: The 10th International Workshop on Frontiers in Handwriting Recognition (IWFHR).

Cireşan, D.C., Meier, U., Gambardella, L.M., Schmidhuber, J., 2011. Convolutional neural network committees for handwritten character classification. In: The 11th International Conference on Document Analysis and Recognition (ICDAR), pp. 1135–1139.

Cireşan, D.C., Meier, U., Schmidhuber, J., 2012. Multi-column deep neural networks for image classification. In: 2012 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp. 3642–3649.

Dalal, N., Triggs, B., 2005. Histograms of oriented gradients for human detection. In: IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp. 886–893.

Das, N., Das, B., Sarkar, R., Basu, S., Kunda, M., Nasipuri, M., 2010. Handwritten Bangla basic and compound character recognition using MLP and SVM classifier. J. Comput. 2 (2), 109–115.

Déniz, O., Bueno, G., Salido, J., la Torre, F.D., 2011. Face recognition using histograms of oriented gradients. Pattern Recognit. Lett. 32 (12), 1598–1603.

Gupta, A., Srivastava, M., Mahanta, C., 2011. Offline handwritten character recognition using neural network. In: 2011 IEEE International Conference on Computer Applications and Industrial Electronics (ICCAIE), pp. 102–107.

Hinton, G.E., Osindero, S., Teh, Y.-W., 2006. A fast learning algorithm for deep belief nets. Neural Comput. 18 (7), 1527–1554.

Jarrett, K., Kavukcuoglu, K., Ranzato, M., LeCun, Y., 2009. What is the best multi-stage architecture for object recognition? In: 2009 IEEE 12th International Conference on Computer Vision (ICCV), pp. 2146–2153.

Kumar, M., Jindal, M., Sharma, R., 2011. k-nearest neighbor based offline handwritten Gurmukhi character recognition. In: 2011 International Conference on Image Information Processing (ICIIP), pp. 1–4.

LeCun, Y., Bottou, L., Bengio, Y., Haffner, P., 1998. Gradient based learning applied to document recognition. In: Proceedings of the IEEE, vol. 86 (11), pp. 2278–2324.

LeCun, Y., Cortes, C., 1998. The MNIST database of handwritten digits.

Lee, S.E., Min, K., Suh, T., 2013. Accelerating histograms of oriented gradients descriptor extraction for pedestrian recognition. Comput. Electr. Eng. 39 (4), 1043–1048.

Liu, C.L., Nakashima, K., Sako, H., Fujisawa, H., 2003. Handwritten digit recognition: benchmarking of state-of-the-art techniques. Pattern Recognit. 36 (10), 2271–2285.

Liu, C.L., Suen, C.Y., 2009. A new benchmark on the recognition of handwritten Bangla and Farsi numeral characters. Pattern Recognit. 42 (12), 3287–3295 N. Front. Handwrit. Recognit.

Lowe, D.G., 2004. Distinctive image features from scale-invariant keypoints. Int. J. Comput. Vis. 60 (2), 91–110.

Mandal, S., Sur, S., Dan, A., Bhowmick, P., 2011. Handwritten Bangla character recognition in machine-printed forms using gradient information and Haar wavelet. In: The International Conference on Image Information Processing (ICIIP), pp. 1–6.

Meier, U., Cireşan, D.C., Gambardella, L.M., Schmidhuber, J., 2011. Better digit recognition with a committee of simple neural nets. In: The 11th International Conference on Document Analysis and Recognition (ICDAR), pp. 1250–1254.

Mitrpanont, J., Imprasert, Y., 2011. Thai handwritten character recognition using heuristic rules hybrid with neural network. In: 2011 Eighth International Joint Conference on Computer Science and Software Engineering (JCSSE), pp. 160–165.

Nopsuwanchai, R., Biem, A., Clocksin, W.F., 2006. Maximization of mutual information for offline Thai handwriting recognition. IEEE Trans. Pattern Anal. Mach. Intell. 28 (8), 1347–1351.

Pal, U., Wakabayashi, T., Kimura, F., 2007. Handwritten Bangla compound character recognition using gradient feature. In: The 10th International Conference on Information Technology (ICIT), pp. 208–213.

Phokharatkul, P., Sankhuangaw, K., Somkuarnpanit, S., Phaiboon, S., Kimpan, C., 2007. Off-line hand written Thai character recognition using ant-miner algorithm. Int. J. Comput. Inf. Syst. Control Eng. 1 (8), 2596–2601.

Rathi, Rakesh, Ravi Krishan Pandey, V.C., Jangid, M., 2012. Offline handwritten Devanagari vowels recognition using KNN classifier. Int. J. Comput. Appl. 49 (23), 11–16.

Schmidhuber, J., 2015. Deep learning in neural networks: an overview. Neural Netw. 61, 85–117.

Schomaker, L.R.B., Vuurpijl, L., 2000. Forensic writer identification: a benchmark data set and a comparison of two systems. Technical Report, University of Nijmegen.

Seo, J., Park, H., 2014. Robust recognition of face with partial variations using local features and statistical learning. Neurocomputing 129 (0), 41–48.

Sun, Y., Zhao, L., Huang, S., Yan, L., Dissanayake, G., 2014. L^2-SIFT: SIFT feature extraction and matching for large images in large-scale aerial photogrammetry. ISPRS J. Photogramm. Remote Sens. 91 (0), 1–16.

Surinta, O., Schomaker, L.R.B., Wiering, M.A., 2012. Handwritten character classification using the hotspot feature extraction technique. In: The 1st International Conference on Pattern Recognition Applications and Methods (ICPRAM), SciTePress, Algarve, Portugal, pp. 261–264.

Surinta, O., Schomaker, L.R.B., Wiering, M.A., 2013. A comparison of feature and pixel-based methods for recognizing handwritten Bangla digits. In: The 12th International Conference on Document Analysis and Recognition (ICDAR), IEEE Computer Society, Washington, DC, pp. 165–169.

van der Maaten, L., 2009. A new benchmark dataset for handwritten character recognition. Technical Report TiCC TR 2009-002, Tilburg University.

Vapnik, V.N., 1998. Statistical Learning Theory. Wiley.