INT J COMPUT COMMUN, ISSN 1841-9836
8(2):220-234, April, 2013.

Improving Offline Handwritten Digit Recognition Using Concavity-Based Features

M. Karic, G. Martinovic

Miran Karic, Goran Martinovic
J. J. Strossmayer University of Osijek
Croatia, 31000 Osijek, Kneza Trpimira 2b
E-mail: miran.karic@etfos.hr,
goran.martinovic@etfos.hr

Abstract:
This paper examines benefits of using concavity-based structural features in recognition of handwritten digits. An overview of existing concavity features is presented and a new method is introduced. These features are used as complementary features to gradient and chaincode features, both among the best performing features in handwritten digit recognition. Two support vector classifiers (SVCs) are chosen for classification task as the top performers in previous works; SVC with radial basis function (RBF) kernel and the SVC with polynomial kernel. For reference, we also used the k-nearest neighbor (k-NN) classifier. Results are obtained on MNIST, USPS and DIGITS datasets. We also tested dataset independency of various feature vectors by combining different datasets. The introduced feature extraction method gives the best results in majority of tests.
Keywords: Complementary features, concavity features, digit recognition, feature extraction, handwritten character recognition, off-line recognition.

1 Introduction

Handwritten digit recognition is an important area of optical character recognition research. Common applications are bank check processing, postal code recognition for mail sorting and recognition of various forms for automated data entry. These applications require high recognition accuracy and speed. Handwritten digit recognition is also often used as a platform for testing performance of classification algorithms.

Typical character recognition process consists of preprocessing, segmentation, feature extraction and classification [1]. This paper deals only with feature extraction and classification. Selection of features and classifiers is vital for performance of a recognition system. In [2] [3] it was concluded that feature extraction is of primary importance in character recognition tasks. Even simple classifiers can give very high recognition accuracy when a well-chosen feature extraction method is used. A better classifier can still be used to improve the recognition accuracy. Combining classifiers is another method used to improve the accuracy; however, we will only discuss single classifier recognition.

Feature extraction is a process for capturing relevant characteristics of a target object (in this case a digit) from an image with a fixed number of feature variables that make a feature vector. It is preferable that the size of a feature vector be as small as possible [4]. Process is sometimes skipped and the classification is performed directly on the raw image data. Numerous types of features for offline handwritten digit recognition exist, ranging from structural, which are based on geometric and topological properties of a digit, to statistical, which are based on digit image statistical properties. Good features should maximize the between-class variance [2] [5].

Classification is a process of assigning new data to a category based on training data in known categories. In this paper, we use a number of human identified digit images split into training and test set. A classifier learns on training images and labels and produces output based on

test images. Output is then compared to test labels to evaluate the classification performance. A good classifier should be able to learn on the training data but maintain the generalization property to be accurate when identifying the test set.

In [2] performance of a range of features and classifiers was tested in handwritten digit recognition tasks. It was concluded that gradient feature [6] and chaincode feature [7], both types of direction features, overall performed best in all tests. Furthermore, adding complementary structural features may improve the accuracy as in [2] [8]. Support vector classifier (SVC) with radial basis function (RBF) kernel gave the highest accuracy.

This paper further investigates how complementary structural features affect overall recognition accuracy. Complementary features incorporate character properties that supplement primary feature vector. One type of studied complementary features in [2] is concavity features. We expand this research with an introduced variation of concavity based complementary features and another variation described in [9]. Gradient and chaincode features are used as primary features. Classification is performed using the support vector classifiers and k-nearest neighbor (k-NN) classifier. New data is also used to test recognition performance.

Datasets used for the experiments are MNIST [10], USPS [11] and DIGITS [12]. The first two are well known datasets in evaluation of handwritten digit recognition and classification algorithms [2] [13] [14], while the last one is relatively unknown [15]. All are divided into standard training and test sets to ensure fair comparison of different classification meethods. Transferring expertise from one dataset to another is an unsolved problem [12] [15] so we also tested how various features perform on data obtained by combining different datasets.

To make this research easy to reproduce and extend, source codes in Matlab m-file format used to extract features and links to used datasets are available online [16].

The remainder of the paper is organized as follows. Section 2 explains primary and complementary feature extraction methods in detail. Section 3 brings a summary of classification methods. Section 4 presents datasets used to test recognition performance and some previous results on these datasets. Section 5 shows the experimental setup. Recognition results are shown and discussed in Section 6. Conclusion is given in Section 7.

2 Feature Extraction

This section presents feature extraction methods used in handwritten digit recognition experiments. First, the primary features are shown, namely gradient and chaincode features. Following are complementary features based on digit image concavity. Two existing and an introduced concavity feature extraction methods are explained. For every method, first a brief explanation of the method is given, and then the setup we used in the experiments.

Source images for feature extraction are binary images. Size of all images matches the size of MNIST dataset images. Other datasets are converted into the format of MNIST dataset. Feature vectors are scaled to values in range $0 - 1$ so that all feature variables would contribute to classification process to the same extent. Scaling is performed separately by feature extraction method, meaning if we use gradient and concavity feature vector, first a maximum value $fmax_1$ of gradient feature vector is found. Gradient feature vector is divided by $fmax_1$. Concavity feature vector is divided by its maximum value $fmax_2$. Feature vectors are then merged.

2.1 Gradient

Gradient features, as in [1] [2] [6], are calculated by using the Sobel operator masks (figure 1) on character image to compute the gradient components on two axes. Grayscale images should be used, but if source images are binary they can simply be converted into pseudogray images,

as in [1] [2] [6]. Gradient strength and direction is computed from gradient components in every character image pixel.

-1	0	1
-2	0	2
-1	0	1

1	2	1
0	0	0
-1	-2	-1

Figure 1: Sobel masks

Gradient vectors are then mapped to, most often, four or eight standard directions. Every vector is decomposed into two components on two nearest standard directions. Figure 2 shows eight standard directions and gradient vector decomposition. Character image is divided into zones by a grid, usually 4×4 or 5×5. The total sum of the component vectors is calculated for each standard direction in a zone. Standard direction intensities for all zones make a feature vector. Different but similar methods for gradient feature extraction can also be found in the literature, as in [9].

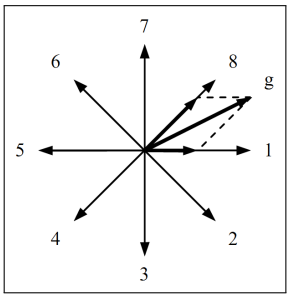


Figure 2: Gradient standard directions and vector decomposition

Since the digit recognition in our experiments is performed on binary images, first an image is converted into a pseudo-gray image. A 3×3 Gaussian lowpass filter with $\sigma = 0.5$ is used to blur binary image. Gradient features are then extracted using eight standard directions and a 5×5 grid. This makes a feature vector of size $5 \times 5 \times 8 = 200$. A transformation on the feature vector, $y = x^{0.5}$, known as Box-Cox transform [17], is carried out to make its distribution closer to the normal distribution.

2.2 Chaincode

Chaincode features, as in [2] [7] are calculated based on a character contour. Every pixel on a contour is assigned a direction code, based on its succeeding pixel's relative position, as in figure 3. There are eight possible directions. Character image is then divided into zones by a grid, usually 4×4 or 5×5. For every zone, number of direction codes for each direction is counted, making a feature vector. Number of features can be halved by summing codes in opposite directions, as in [2].

Chaincode feature extraction is performed directly on binary digit images. 5×5 grid and information in eight directions was used to form a vector of $5 \times 5 \times 8 = 200$ feature variables. Box-Cox transformation $y = x^{0.5}$, as in [17], is used on the feature vector to make its distribution closer to the normal distribution.

2.3 Concavity

Concavity features are used primarily as complementary features since they contain only a limited amount of information but can improve recognition accuracies of other feature extraction

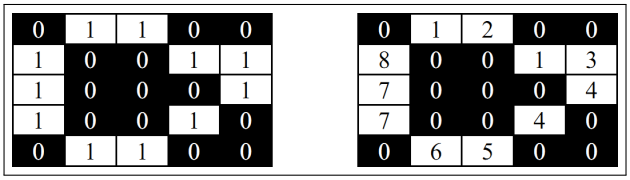


Figure 3: Example of a contour (a) and its chaincode values (b)

methods. They are structural features based on measurements of character concavities. This paper considers three different concavity feature extraction methods; two existing and one new. The first method by Favata et al. is explained in [9], we will label it *conc1* for future reference. The second method by Liu et al. is shown in [2] and is labeled *conc2*. The introduced method is labeled *conc3*. All concavity features are extracted from binary images. Below is a short overview of existing methods and a more extensive explanation of the new method.

Method *conc1* uses a star-like operator that shoots rays in eight directions. It is observed what each ray hits, a character or image border. This operator is applied to every pixel on a digit image. Unfortunately, there is no detailed explanation of the method that would allow identical reproduction, however we believe that our reproduction should produce similar results as it utilizes the same idea. Authors themselves say that they presented one particular implementation of their philosophy and that others are possible.

Image is divided into zones by a 5×5 grid, and for every zone total number of border hits for each direction is counted. Similar procedure is used in gradient feature extraction. To reduce the number of features, diagonal hits are divided among neighboring horizontal and vertical direction counts. Total number of features is $5 \times 5 \times 4 = 100$. Figure 4a shows *conc1* feature extraction.

Method *conc2* measures the distance from character convex hull to character pixels. Only horizontal distances are calculated, both from the left and from the right. Figure 4b shows *conc2* feature extraction. In [2] it is used in conjunction with crossings (*crs*) feature extraction method. This method counts the number of transitions from black to white pixels on a binary image for every row. Total number of features depends on image size, and for *conc2* it is equal to two times the image height in pixels, while for *crs* it is equal to image height.

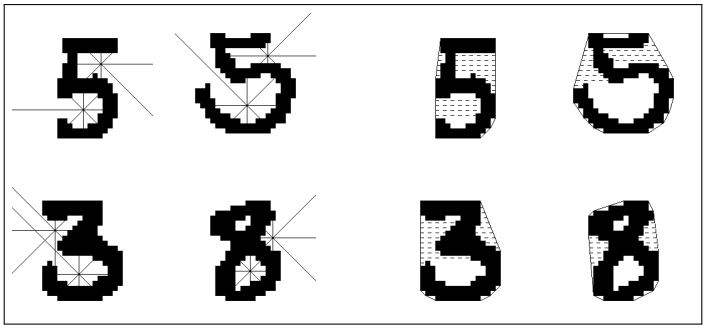


Figure 4: *conc1* (a) and *conc2* (b) feature extraction

We tested by using only *conc2,* and by using both *conc2* and *crs* features. Since all digit images we used areactually 20×20 pixels centered by mass in a 28×28 pixel frame, we performed *conc2* feature extraction on these 20×20 pixels. This gave us a feature vector of 40 variables. *crs* features were extracted from the whole 28×28 image, giving additional 28 features.

The proposed concavity feature extraction method, *conc3*, is based on measurements of char-

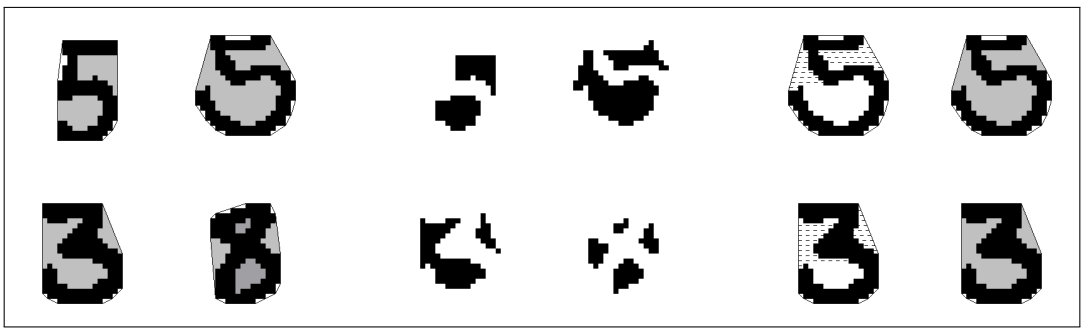
acter concavity regions in a binary image. First the convex hull of a character is calculated using a method described in [18]. By subtracting character image from the hull, concavity regions are obtained. Position of the center of mass, width, height, and the area of each outer region, i.e. on the convex hull border, are calculated. For the inner regions, the center of mass position and the area are calculated plus one feature variable denoting that an inner concavity exists. There are 5 feature variables per outer region and 4 feature variables per inner region. Number of observed outer and inner regions must be defined in advance.

Since we use these features for handwritten digit recognition, only two inner regions are possible and occur for number eight, thus we observe a maximum of two inner regions. After some experimenting we concluded that a maximum of five outer regions give best results. This gives a feature vector of $2 \times 4 + 5 \times 5 = 33$ variables. Feature variables are sorted by area size, separately for inner and outer regions. Four measurements for an inner region with the largest area are the first four variables of a feature vector. Measurements of the next inner region by area make following four variables. Measurements of the outer regions follow, again starting from a region with the largest area. If any of the regions does not exist, feature variables are filled with zeros, only position is filled with values 0.5, 0.5 indicating the character center. To minimize impact of noise and errors incurred during image retrieval, regions under a certain threshold are cut off. Figure 5 shows the *conc3* feature vector variables, while figure 6 illustrates the feature extraction process.

Feature vector						
Inner regions		Outer regions				
1	2	1	2	3	4	5
Cx_{i1}	Cx_{i2}	Cx_{o1}	Cx_{o2}	Cx_{o3}	Cx_{o4}	Cx_{o5}
Cy_{i1}	Cy_{i2}	Cy_{o1}	Cy_{o2}	Cy_{o3}	Cy_{o4}	Cy_{o5}
$area_{i1}$	$area_{i2}$	w_{o1}	w_{o2}	w_{o3}	w_{o4}	w_{o5}
$exist_{i1}$	$exist_{i2}$	h_{o1}	h_{o2}	h_{o3}	h_{o4}	h_{o5}
		$area_{o1}$	$area_{o2}$	$area_{o3}$	$area_{o4}$	$area_{o5}$

Figure 5: *conc3* feature vector. Cx and Cy are region center of mass coordinates, w and h are region width and height.

It can be observed that *conc2* features will not take into account the whole concavity of a character. This can be seen on figure 6c. *conc3* features have the advantage of using the whole concavity, which is the basis for the assumption that our method will give better results.



(a) (b) (c)

Figure 6: *conc3* feature extraction: (a) determining inner and outer regions, (b) extracted regions, (c) comparison of *conc2* (left) and *conc3* (right) features

3 Classification

Support vector machines with radial basis function (SVM-rbf) kernel and polynomial kernel (SVM-poly) are selected for classification. SVMs often give best results in digit recognition, as in [2]. k-NN classifier is included as a simple classifier for comparison with SVMs. More detail on classifiers is presented below.

3.1 Support vector classifiers

In general, support vector machines (SVMs) solve binary classification problems. Multi class classification is accomplished by combining multiple binary SVMs [2]. For SVMs, a solution to an optimization problem is required, defined as follows [8] [13] [19]:

$$\min_{\omega,b,\xi}\left\{\frac{1}{2}\omega^T\omega + C\sum_{i=1}^{l}\xi_i\right\}, \tag{1}$$

$$y_i\left(\omega^T\phi\left(x_i\right)+b\right) \geq 1-\xi_i, \xi_i \geq 0, i=1,...,l,$$

where (x_i, y_i) are training set instance-label pairs, C is the regularization parameter, w is the vector of coefficients, b a constant, ξ_i are parameters for handling nonseparable data (inputs) and ϕ maps input into higher-dimensional space. Usually the following problem is solved:

$$\min_{\alpha}\left\{\frac{1}{2}\alpha^T Q\alpha - e^T\alpha\right\}, \tag{2}$$

$$y_i^T\alpha = 0, 0 \leq \alpha_i \leq C, i=1,...,l,$$

where e is the vector of all ones, Q is an l by l positive semidefinite matrix, $Q_{ij} \equiv y_i y_j K\left(x_i, x_j\right)$ are training set instance-label pairs and $K\left(x_i, x_j\right) \equiv \phi\left(x_i\right)^T\phi\left(x_j\right)$ is the kernel function. We used two kernels, radial basis function (RBF) (3) and polynomial (4) kernel:

$$K\left(x_i, x_j\right) \equiv \exp\left(-\gamma\left\|x_i - x_j\right\|^2\right), \tag{3}$$

$$K\left(x_i, x_j\right) \equiv \left(x_i^T x_j\right)^d. \tag{4}$$

To find optimal values of the variance parameter (γ) of the RBF kernel and the cost parameter C of the SVM we used a simple grid search with values 2^i where i is in range -15 to 3 for γ, and -3 to 15 for C, with the minimum step value of 0.1. Similar procedure is used for the polynomial kernel of the second degree. For all SVM classification tasks we used the LIBSVM library [19].

3.2 k-NN classifier

The k-nearest neighbor algorithm (k-NN) is a method for classification based on the nearest training objects in the feature space. When k=1, the class of the nearest training objects becomes the class of the test object. When k>1 the class of the test object is determined by the majority vote of its neighbors. A weighted version of the algorithm is used in this paper. Each of the neighbors are assigned a weight equal to inverse of distance to test object. This ensures that closer neighbours contribute more to the decision than distant neighbours.

4 Datasets

This section presents the tested datasets. All used datasets contain grayscale images of handwritten digits. Since binary images are required, we converted images from all datasets to binary form using Otsu's global threshold method [20] [21].

4.1 MNIST

MNIST dataset is a well known dataset for handwritten digit recognition, created by Yann LeCun [10]. It has 60000 training samples and 10000 test samples. All images are size 28×28 pixels with 256 levels of gray. Actual size of digit images is 20×20 pixels, centered in a larger image using center of mass. The datasets are available online, along with a list of best performing methods and their recognition accuracies [22]. Several binarized samples from the MNIST dataset are shown in figure 7a.

4.2 USPS

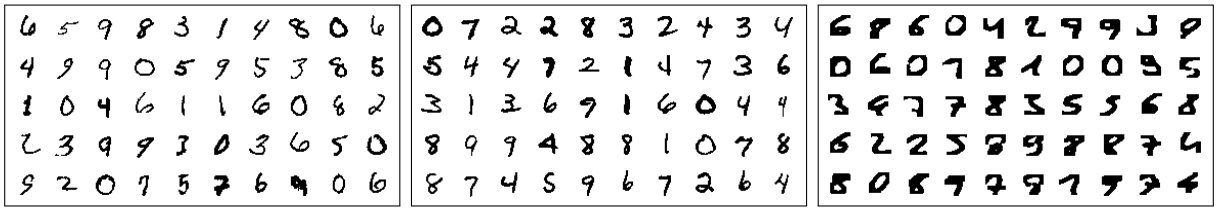
USPS dataset is a US Postal handwritten digit dataset with images obtained from envelopes [11]. It has 7291 training samples and 2007 test samples. Digits are scaled to fit in 16×16-pixel images with 256 levels of gray. Some experimental results on USPS dataset are available online [23]. Several binarized samples from the USPS dataset are shown in figure 7b.

4.3 USPS-r

A modified version of USPS dataset, labeled USPS-r, is also used, since the distribution of original USPS is uneven on training and test sets [24]. The modified version is created by merging both sets, reshuffling and then randomly dividing the full set to a new training and test set of equal size. Each set has 4649 samples. The same dataset as in [24], available online, was used in experiments.

4.4 DIGITS

DIGITS dataset is a less known dataset for handwritten digit recognition, described in [12]. It has 1893 training samples and 1796 test samples. Digits are prepared in a similar format to USPS dataset, with16×16-pixel images and 256 levels of gray. Experimental results on this dataset can be found in [12]. Several binarized sample images from the DIGITS dataset are shown in figure 7c.



(a) (b) (c)

Figure 7: Samples from MNIST (a), USPS (b) and DIGITS (c) datasets

4.5 COMBINED

Another dataset is used, derived by combining MNIST, USPS-r and DIGITS datasets. Its training set actually consists of the equal number of instances from the three datasets. DIGITS training set has the smallest number of instances therefore the equal amount of instances is randomly taken from training sets of each of the other two datasets, creating two shorter training sets, labelled MNIST-s and USPS-s. COMBINED dataset consists of MNIST-s, USPS-s and DIGITS training sets. 1893 instances are taken from each dataset, giving 5679 instances total.

Experiments are conducted by training on COMBINED, MNIST-s or USPS-s sets and testing on the test set of MNIST, USPS-r or DIGITS datasets. This allows us to compare recognition accuracy on datasets with equal training set size, when using MNIST-s and USPS-s. Using COMBINED set allows us to verify whether recognition accuracy can be increased by learning on instances from another dataset, and what type of features are performing best. A similar method is used in [15], where training on training sets of two datasets and testing on the test set of the third was performed. This gave very poor recognition accuracies, often an order of magnitude lower than when training on the training set of the same dataset. This might produce unreliable results therefore we decided to experiment by adding instances.

5 Experimental setup

Handwritten digit recognition performance was tested on four different datasets (MNIST, two versions of USPS and DIGITS) using four classifiers and ten feature vectors. This gives a total of 40 recognition accuracies for each dataset. Support vector classifiers with RBF kernel and second degree polynomial kernel are used for classification as described earlier. k-NN classifier is also used for comparison. Recognition accuracy is shown for the case $k = 1$, i.e. 1-NN, and for the best result for all values of $k = 1, 2 \ldots 11$, marked k-NN, using the method described earlier.

All datasets are first converted to binary form because the features used in experiments are extracted from binary images. Images are then converted to MNIST format. Image area containing a digit is scaled to size 20×20 pixels, keeping the aspect ratio. This scaled image is then placed on a frame of 28×28 pixels, positioning the digit center of mass in the center of the frame.

Feature vectors are defined to contain one primary feature which is combined with complementary features. Five vectors use gradient and five chaincode as primary feature extraction method. In addition, four out of each of the five vectors contain complementary features, *conc1*, *conc2*, *conc2* with *crs* and *conc3*, while one has no complementary features. These are explained earlier. Precise description of the feature vectors is given in table 1. Feature vectors *e-grg*, *e-blr* and *e-mul* exist in [2] and have the same labels.

Feature vector	Features extraction methods	Size
e-grg	gradient	200
e-grc1	gradient + *conc1*	300
e-grc2	gradient + *conc2*	240
e-grc2c	gradient + *conc2* + *crs*	268
e-grc3	gradient + *conc3*	233
e-blr	chaincode	200
e-blc1	chaincode + *conc1*	300
e-blc2	chaincode + *conc2*	240
e-mul	chaincode + *conc2* + *crs*	268
e-blc3	chaincode + *conc3*	233

Table 1: Feature vectors

Following experiments on the four datasets, additional experiments with reduced training sets are executed. This allowed us to investigate the impact of training set size on the recognition accuracy depending on the feature vector. Also we can compare recognition accuracies on datasets when the training set sizes are equal. In addition, merging these reduced training sets allowed us to verify how the recognition accuracy is affected for different feature vectors when

we add training instances which are obtained using a different image acquisition process, in this case belonging to another dataset.

As defined earlier, COMBINED, MNIST-s and USPS-s sets are used for these experiments. Results when training on MNIST-s set and testing on MNIST test set, and also when training on USPS-s and testing on USPS-r test set are obtained. COMBINED set is used for training in combination with test sets of MNIST, USPS-r and DIGITS datasets. Results obtained using COMBINED training set can be compared to results when using MNIST-s, USPS-s and DIGITS training sets to verify how additional instances from other datasets affect recognition performance.

6 Experimental results and discussion

The experimental results in form of error rates are presented in this section, followed by our observations and discussion. Dataset used for an experiment is given in the title of every table. In the table rows the names of the feature vectors described earlier are listed, while the columns contain the used classifiers. Shaded results are obtained using the introduced method, while the best results by category are marked with bold letters. By different category we consider a different primary feature, or a different classifier. More importance is given to analysis of SVM classifier results, while the 1-NN and k-NN results are presented as an indication of SVM classifier superiority.

6.1 Results on the individual datasets

Table 2 shows the results obtained on the MNIST dataset. The best recognition accuracy, i.e. lowest error rate (0.67%), is achieved using the *e-grc3* vector and the SVM-rbf classifier. It is also the best attained recognition accuracy in the article. Immediately behind is the accuracy of the vector *e-grg* (error rate 0.68%) containing only the primary feature, gradient, and no complementary features. Other vectors in the category achieve lower accuracies. Relationships among the vectors when using the SVM-poly classifier remained approximately the same, with slightly lower accuracies, only the vector *e-grc3* has an even greater advantage over the other vectors. Vectors that use chaincode primary feature achieved lower recognition accuracies. Vector *e-mul* has the advantage over the other vectors in its category for both SVM classifiers (error rate 1.00 for SVM-rbf). Vector *e-blc1* is second and *e-blc3* third by accuracy.

Feature vector	1-NN	k-NN	SVM-rbf	SVM-poly
e-grg	1.62	1.24	0.68	0.85
e-grc1	1.67	1.36	0.72	0.96
e-grc2	1.61	1.25	0.74	0.87
e-grc2c	1.45	**1.20**	0.78	0.92
e-grc3	**1.43**	1.26	**0.67**	**0.75**
e-blr	4.17	3.52	1.65	2.01
e-blc1	3.45	2.80	1.04	1.38
e-blc2	3.24	2.88	1.21	1.41
e-mul	2.72	2.47	**1.00**	**1.26**
e-blc3	**2.68**	**2.35**	1.11	1.44

Table 2: Error rates (%) on the MNIST dataset

Table 3 shows the results obtained on the USPS dataset. The best recognition accuracy is once again achieved using the *e-grc3* vector and the SVM-rbf classifier. Error rate of 2.39% is the lowest among results reported in [23] using a single classifier. On top of that, we used binary images for training and testing which puts our method in a disadvantage. Using the SVM-poly classifier, *e-grc3* and *e-grg* (2.64%) have equal error rates, while other vectors have higher error rates, i.e. decreased performance of the primary feature. Vectors with chaincode primary feature give higher error rates with all classifiers, with vector *e-blc3* performing best.

Feature vector	1-NN	k-NN	SVM-rbf	SVM-poly
e-grg	**3.29**	**3.14**	2.64	**2.64**
e-grc1	3.44	3.39	2.69	2.79
e-grc2	3.84	3.39	2.54	2.84
e-grc2c	3.79	3.44	2.54	2.79
e-grc3	3.39	3.34	**2.39**	**2.64**
e-blr	6.48	5.23	3.14	3.49
e-blc1	5.73	4.88	3.04	3.44
e-blc2	5.98	5.23	2.84	3.19
e-mul	5.93	4.93	2.89	3.39
e-blc3	**5.03**	**4.53**	**2.69**	**2.99**

Table 3: Error rates (%) on the USPS dataset

Table 4 shows the results obtained on the USPS-r dataset. Feature vector *e-grc3* and SVM-rbf classifier give the best recognition accuracy on this dataset as well (error rate 1.33%). Results are similar to results on USPS dataset, but with significantly higher recognition accuracies. When using the SVM-poly classifier, vectors with complementary features do not increase the error rate of the primary feature. The vectors *e-grc3* and *e-blc3* still give the highest recognition accuracies.

Feature vector	1-NN	k-NN	SVM-rbf	SVM-poly
e-grg	2.19	**1.87**	1.44	1.72
e-grc1	2.39	1.98	1.42	1.66
e-grc2	2.37	2.00	1.53	**1.63**
e-grc2c	2.30	2.24	1.42	1.66
e-grc3	**2.15**	2.00	**1.33**	**1.63**
e-blr	3.53	3.40	2.26	2.60
e-blc1	**3.20**	3.20	1.96	2.26
e-blc2	3.44	3.44	2.00	2.30
e-mul	3.51	3.18	1.94	2.24
e-blc3	3.25	**3.12**	**1.89**	**2.11**

Table 4: Error rates (%) on the USPS-r dataset

Table 5 shows the results obtained on the DIGITS dataset. Error rates on this dataset are significantly higher than on other tested datasets. Complementary features increase recognition accuracy in all tests. For vectors with gradient primary feature, *e-grc1* gives the best performance overall (error rate 4.73%), *e-grc2c* is second (4.79%) and *e-grc2* and *e-grc3* with equal error rate third (4.90%). Among vectors with chaincode primary feature, *e-blc1* gives the lowest error

rate when SVM-poly classifier is used (4.90%). With SVM-rbf classifier *e-mul* vector achieves the lowest error rate (5.23%). Vector *e-blc3* has the lowest average error rate when both SVM classifiers are taken into account.

Feature vector	1-NN	k-NN	SVM-rbf	SVM-poly
e-grg	7.63	6.46	5.07	5.46
e-grc1	7.68	6.90	**4.73**	**4.90**
e-grc2	7.57	6.51	4.90	5.23
e-grc2c	7.35	6.57	4.79	5.07
e-grc3	**7.18**	**6.18**	4.90	5.35
e-blr	9.80	9.24	6.12	6.51
e-blc1	8.85	8.85	5.51	**5.51**
e-blc2	**8.52**	8.35	5.51	6.18
e-mul	**8.52**	**7.96**	**5.23**	6.12
e-blc3	10.86	8.13	5.35	5.58

Table 5: Error rates (%) on the DIGITS dataset

Overall, according to expectations, SVM-rbf classifier gives the best recognition performance, followed by SVM-poly, while 1-NN and k-NN achieve significantly lower accuracy. The vector *e-grc3* gives the highest accuracy in the majority of the tests, except for the DIGITS dataset where vector *e-grc1* achieved higher accuracy. In addition, *e-grc3* is the only vector improving on the results of the primary feature vector *e-grg* in all tests, while *e-grc1*, *e-grc2* i *e-grc2c* achieved higher error rates than *e-grg* on several occasions. Figure 8 gives error rates on four datasets using the SVM-rbf classifier for vectors containing gradient primary feature. It can be seen that concavity features improve recognition accuracy, with vector *e-grc3*, containing proposed concavity features, giving best recognition performance. Vectors using the chaincode primary feature achieve lower recognition accuracies. Among these vectors, the best results are achieved using the proposed vector *e-blc3* and vector *e-mul*.

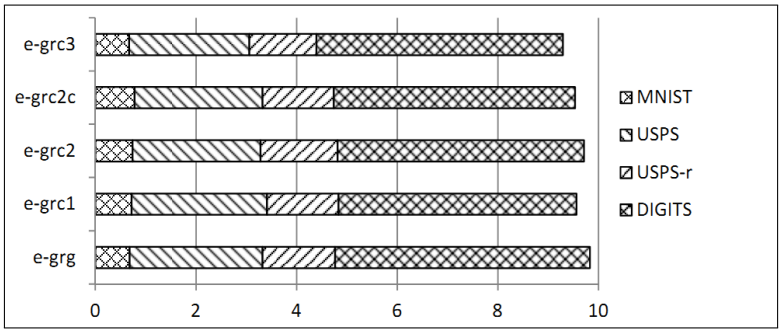


Figure 8: Error rates on four datasets for five feature vectors using SVM-rbf classifier

6.2 Results on combined datasets

The results of recognition on the MNIST test set, when training on the reduced training set MNIST-s, are shown in table 6. Compared to the results obtained by training on the full MNIST training set, recognition accuracies are lower. The use of complementary features greatly improves the recognition accuracy, with feature vector *e-grc3* performing best (error rate 1.66% using SVM-rbf). Vector *e-blc3* achieves lower error rate than *e-mul* when reduced training set

is used. When training on the COMBINED dataset, best results are still obtained by using the vector *e-grc3*, as shown in table 7. Additional training images improved the recognition accuracy of *e-grc3* (error rate 1.54% using SVM-rbf). Vector *e-mul* now has higher recognition accuracy than vector *e-blc3*, however it has not increased.

Feature vector	1-NN	k-NN	SVM-rbf	SVM-poly
e-grg	3.91	3.56	2.06	2.42
e-grc1	4.40	3.80	2.03	2.29
e-grc2	3.79	3.30	1.87	2.11
e-grc2c	3.70	3.30	1.80	2.16
e-grc3	**3.18**	**2.75**	**1.66**	**1.92**
e-blr	8.03	7.30	3.55	4.38
e-blc1	7.07	6.43	3.01	3.44
e-blc2	6.18	5.84	2.81	3.28
e-mul	**5.48**	5.22	2.59	**3.10**
e-blc3	5.51	**4.82**	**2.53**	3.12

Table 6: Error rates (%) on the MNIST test set, trained on the MNIST-s training set

Feature vector	1-NN	k-NN	SVM-rbf	SVM-poly
e-grg	4.05	3.51	1.80	2.26
e-grc1	4.45	3.87	1.82	2.45
e-grc2	3.89	3.27	1.72	2.08
e-grc2c	3.84	3.18	1.70	2.11
e-grc3	**3.29**	**2.75**	**1.54**	**1.94**
e-blr	8.07	7.07	3.66	4.39
e-blc1	7.00	6.33	2.81	3.47
e-blc2	6.09	5.64	2.96	3.39
e-mul	5.44	4.82	**2.53**	**3.35**
e-blc3	**5.14**	**4.52**	2.60	3.41

Table 7: Error rates (%) on the MNIST test set, trained on the COMBINED training set

Table 8 shows the results obtained on the USPS-r test set, when training on the reduced training set USPS-s. Feature vectors *e-grc3* and *e-blc3* achieve best recognition accuracies in their categories in all tests. Accuracy of the vector *e-blc3* is only matched by *e-mul* when using the SVM-rbf classifier. The lowest error rate is achieved by the vector *e-grc3* (2.06%). Vectors with complementary features *e-grc1*, *e-grc2* and *e-grc2c* fail to decrease the error rate of the primary feature vector *e-grg*. Introducing additional training images overall reduced the error rates considerably, as shown in table 9. Feature vector e-*grc3* is still unmatched (error rate 1.74%), while other vectors with complementary features again fail to decrease the error rate of *e-grg* in their categories. Vector *e-blc3* is unmatched among vectors using chaincode primary feature, decreasing error rates more then other vectors when additional training images are added.

Feature vector	1-NN	k-NN	SVM-rbf	SVM-poly
e-grg	3.18	2.80	2.15	2.54
e-grc1	3.36	3.03	2.26	2.41
e-grc2	3.25	2.77	2.15	2.50
e-grc2c	3.48	3.08	2.15	2.43
e-grc3	**2.90**	**2.65**	**2.06**	**2.28**
e-blr	4.88	4.52	2.95	3.05
e-blc1	4.60	4.28	2.88	3.05
e-blc2	4.93	4.56	2.77	2.99
e-mul	4.80	4.43	**2.71**	2.88
e-blc3	**4.17**	**3.94**	**2.71**	**2.86**

Table 8: Error rates (%) on the USPS-r test set, trained on the USPS-s training set

Feature vector	1-NN	k-NN	SVM-rbf	SVM-poly
e-grg	3.16	2.65	1.85	2.06
e-grc1	3.46	2.97	1.89	2.13
e-grc2	3.23	2.67	1.87	2.22
e-grc2c	3.33	2.93	1.85	2.04
e-grc3	**2.99**	**2.58**	**1.74**	**1.91**
e-blr	4.99	4.47	2.65	2.93
e-blc1	4.75	4.24	2.47	2.73
e-blc2	5.08	4.34	2.60	2.80
e-mul	5.01	4.32	2.43	2.60
e-blc3	**4.17**	**3.61**	**2.19**	**2.47**

Table 9: Error rates (%) on the USPS-r test set, trained on the COMBINED training set

Since all images of the DIGITS dataset training set are included in the COMBINED training set, error rates on the DIGITS dataset can be compared to error rates on the DIGITS test set, trained on the COMBINED training set, shown in table 10. Introducing additional images to training set increased error rates on this test set. Possible explanation for this outcome is a different methodology used for retrieving data of DIGITS dataset and the other two datasets. Significantly higher error rates on the DIGITS dataset than on the other two datasets also indicate difference in the data format. Vectors with complementary features are not successful in increasing the recognition accuracy of the gradient primary feature vector *e-grg*. Among vectors with chaincode primary feature, complementary features increase the recognition accuracy, with the vector *e-mul* giving the best recognition accuracy.

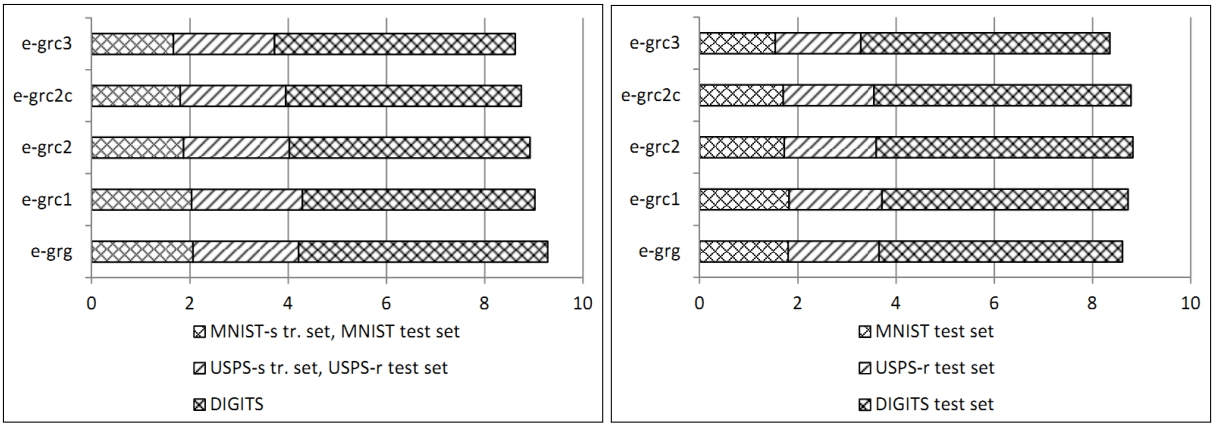
Feature vector	1-NN	k-NN	SVM-rbf	SVM-poly
e-grg	7.46	6.40	**4.96**	**5.62**
e-grc1	7.52	6.63	5.01	**5.62**
e-grc2	7.46	6.29	5.23	5.79
e-grc2c	7.24	6.24	5.23	5.85
e-grc3	**7.13**	**5.90**	5.07	**5.62**
e-blr	9.13	8.57	5.90	7.02
e-blc1	8.57	8.57	5.74	**6.46**
e-blc2	8.02	7.41	5.57	6.63
e-mul	**7.96**	**7.35**	**5.46**	6.74
e-blc3	10.19	7.68	5.79	**6.46**

Table 10: Error rates (%) on the DIGITS test set, trained on the COMBINED training set

Best performance of the SVM-rbf classifier in combination with the vector *e-grc3* was shown in this series of experiments as well. For every vector, best recognition accuracy is given when using the SVM-rbf classifier. The vector *e-grc3* gives the best recognition accuracy in all experiments except on the DIGITS dataset. Taking into account experiments on individual and combined datasets, the vector *e-grc3* gives the highest recognition accuracy on seven out of nine different experimental setups. Graphical representation of the results obtained when training on reduced training sets is given in figure 9a and when training on combined training sets in figure 9b. It can be concluded that the vector *e-grc3* gives the best results overall. Vectors using the chaincode primary feature achieve lower recognition accuracies than vectors using the gradient primary feature. The vector *e-blc3* and vector *e-mul* give lowest error rates in their categories.

7 Conclusion

Experimental results showed that complementary features can significantly improve recognition performance. The proposed concavity feature extraction method in conjunction with gradient features gave the highest recognition accuracy in majority of experiments. The method worked well with chaincode features as well, being one out of two top performers. It also has the



(a) (b)

Figure 9: Error rates for five feature vectors using SVM-rbf classifier (a) on three datasets with equal (reduced) training set size and (b) on datasets with training on COMBINED set

lowest feature count among observed complementary features, which lowers computational cost of classification. Experiments using reduced training sets showed that the proposed concavity method outperforms other observed approaches making it useful for applications requiring use of a small training set. Adding training instances from another dataset reflected on the recognition accuracy differently for different datasets. Accuracy was increased on two datasets and decreased on one, indicating that learning process is sensitive to small differences in image retrieval and preprocessing. Overall, the proposed method achieved the best performance.

Acknowledgements

This work was supported by research project grant No. 165-0362980-2002 from the Ministry of Science, Education and Sports of the Republic of Croatia.

Bibliography

[1] C.-L. Liu, K. Nakashima, H. Sako, H. Fujisawa, Handwritten digit recognition: investigation of normalization and feature extraction techniques, *Pattern Recognition*, 37(2):265-279, 2004.

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

[3] M. H. Nguyen, F. de la Torre, Optimal feature selection for support vector machines, *Pattern Recognition*, 43(3):584-591, 2010.

[4] U. Kressel, J. SchĂźrmann, Pattern classification techniques based on function approximation, *Handbook of Character Recognition and Document Image Analysis*, 49-78, 1997.

[5] B. P. Chacko, P. Babu Anto, Comparison of Statistical and Structural Features for Handwritten Numeral Recognition, *Proc. of the Int. Conf. on Computational Intelligence and Multimedia Applications (ICCIMA 2007)*, Washington, DC (USA), 296-300, 2007.

[6] H. Liu, X. Ding, Handwritten Character Recognition Using Gradient Feature and Quadratic Classifier with Multiple Discrimination Schemes, *Proc. of the Eighth Int. Conf. on Document Analysis and Recognition (ICDAR '05)*, Washington, DC (USA), 19-25, 2005.

[7] O. D. Trier, A. K. Jain, T. Taxt, Feature Extraction Methods for Character Recognition - A Survey, *Pattern Recognition*, 29(4):641-662, 1996.

[8] G. Vamvakas, B. Gatos, I. Pratikakis, N. Stamatopoulos, A. Roniotis, S. J. Perantonis, Hybrid off-line OCR for isolated handwritten Greek characters, *Proc. of the Fourth IASTED Int. Conf. on Signal Processing, Pattern Recognition, and Applications*, Innsbruck (Austria), 197-202, 2007.

[9] J. Favata, G. Srikantan, S. Srihari, Handprinted character/digit recognition using a multiple feature/resolution philosophy, *Fourth International Workshop on Frontiers in Handwriting Recognition*, Taipei (Taiwan), 67-70, 1994.

[10] Y. LeCun, L. Bottou, Y. Bengio, P. Haffner, Gradient-Based Learning Applied to Document Recognition, *Proceedings of the IEEE*, 86(11):2278-2324, 1998.

[11] J. J. Hull, A Database for Handwritten Text Recognition Research, *Pattern Analysis and Machine Intelligence*, 16(5):550-554, 1993.

[12] A. K. Seewald, Digits - A Dataset for Handwritten Digit Recognition, *Austrian Research Institut for Artificial Intelligence Technical Report*, Vienna (Austria), 2005.

[13] C. Cortes, V. Vapnik, Support-Vector Networks, *Machine Learning*, 20(3):273-297, 1995.

[14] L. Van der Maaten, A New Benchmark Dataset for Handwritten Character Recognition, *Tilburg University Technical Report*, 2009.

[15] A. K. Seewald, On the Brittleness of Handwritten Digit Recognition Models, *Technical Report, Seewald Solutions*, Vienna (Austria), 2009.

[16] M. Karic, Concavity paper source code. [Online] Cited 2011-08-30. Available at: http://www.etfos.hr/ mkaric/conc.

[17] R. V. D. Heiden, F. C. A. Gren, The Box-Cox metric for nearest neighbor classification improvement, *Pattern Recognition*, 30(2):273-279, 1997.

[18] C. B. Barber, D. P. Dobkin, H. T. Huhdanpaa, The Quickhull Algorithm for Convex Hulls, *ACM Trans. on Mathematical Software*, 22(4):469-483, 1996.

[19] C.-C. Chang, C.-J. Lin, LIBSVM: a library for support vector machines, *ACM Trans. on Intelligent Systems and Technology*, 2(3): 1-39, 2012. Software available at http://www.csie.ntu.edu.tw/ cjlin/libsvm.

[20] N. Otsu, A Threshold Selection Method from Gray-Level Histograms. *IEEE Trans. on Systems, Man and Cybernetics*, 9(1):62-66, 1979.

[21] M. R. Gupta, N. P. Jacobson, E. K. Garcia, OCR binarization and image pre-processing for searching historical documents, *Pattern Recognition*, 40(2):389-397, 2007.

[22] Y. LeCun, The MNIST database of handwritten digits. [Online] Cited 2011-08-30. Available at: http://yann.lecun.com/exdb/mnist.

[23] D. Keysers, Experimental results on the USPS database. [Online] Cited 2011-08-30. Available at: http://www-i6. informatik.rwth-aachen.de/ keysers/Pubs/SPR2002/node10.html.

[24] C. E. Rasmussen, C. K. I. Williams, *Gaussian Processes for Machine Learning*, 2nd ed., The MIT Press, 2006.