Optical Character Recognition of Seven–segment Display Digits Using Neural Networks

Ines Bonačić, Tomislav Herman, Tomislav Krznar, Edin Mangić, Goran Molnar and Marko Čupić
Faculty of Electrical Engineering and Computing
University of Zagreb
Unska 3, 10.000 Zagreb, Croatia
Phone: (+385) 1 6129 548
E-mail: {ines.bonacic | tomislav.herman | tomislav.krznar | edin.mangic | goran.molnar2 | marko.cupic} @ fer.hr

Abstract - In this work, we present a neural networks committee for optical character recognition of seven-segment display digits. The aforementioned digit writing convention restricts the general handwriting recognition problem into a task that can be tackled using an automated approach. Numerous practical applications are available for this type of digit recognition: various forms, written exams etc. We consider three different digit recognition techniques. An appropriate feed-forward neural network is devised for each technique. We use two different methods to determine the optimal topology of neural networks: (1) a traditional, manual approach, and (2) automated topology optimizing system, based on the genetic algorithm. To further improve the performance, a committee of neural networks is developed. In this paper, we elaborate the early results of the character recognition system, based on the devised neural networks committee.

I. INTRODUCTION

The possibility of automated recognition of handwritten text would significantly increase productivity of many types of institutions. Despite intensive research, the problem of efficient and dependable *optical character recognition* (OCR) of handwritten text is still an open issue [2].

We research a problem of the optical character recognition of seven-segment display digits, which is a restriction of the general handwriting recognition problem. Seven-segment display digits are found on calculators, digital watches, as well as video and audio equipment. They are a widely used, and an easily recognizable writing convention. Currently, our system only supports digits from 0 to 9, but the technique is applicable to any other set of symbols that can be represented using a seven-segment template (Fig. 1). Although the restriction we introduce makes digit writing much slower than normal handwriting, it significantly reduces the complexity of automated recognition. This writing convention is not intended for large texts, but instead, to facilitate recognition of large number of copies of documents containing a limited amount of numerical data. Examples of such documents are statistical forms, written exams, and others, where a few numbers per page need to be entered by hand.

The described OCR system has several potential applications, e.g. automated grading of written exams in schools or universities. Using the proposed system, students are expected to write their identification number using seven-segment notation on forms prepared for OCR (Fig. 2). Such a system would make grading of written exams possible in a purely automated way, thus alleviating the staff from the labour intensive task of manual exam grading.

The rest of this paper is organized as follows. In Section II, we give a definition of the seven-segment digit OCR problem, and the methodology of the data-set acquisition. In Section III, we elaborate our approach. In Section IV, we discuss the preprocessing methods. Section V describes the early results of our system and our plans for future work. In Section VI, we present the conclusion.

II. OCR OF SEVEN-SEGMENT DISPLAY DIGITS

A. Definition of the Problem

We define the problem of the optical character recognition of a seven-segment display *digit* as a problem of classification of the digit image into one of ten classes, where each of the classes corresponds to a digit of the decimal numeral system. The problem of recognizing a seven-segment digit *number* is to translate the image of the number into the corresponding numerical value. This task is performed by solving the OCR problem of each of the number's digits.

The end users inscribe the digits in the empty forms. Each form consists of *number fields*: areas of documents where numbers to be recognized are positioned, placed in a defined area of the document. Each number field contains a defined number of *digit fields.* A digit field is a scheme of seven (approximately) rectangular segments, arranged into a well known “eight” pattern, as in Fig. 1. We define the *digit scheme* as a layout of seven segments of a digit type. This scheme contains: (1) information about the exact position and size of the digit, and (2) dimensions and

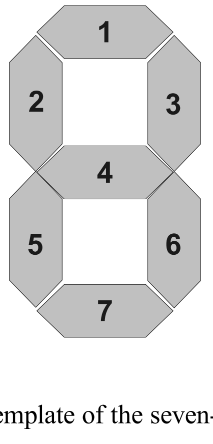


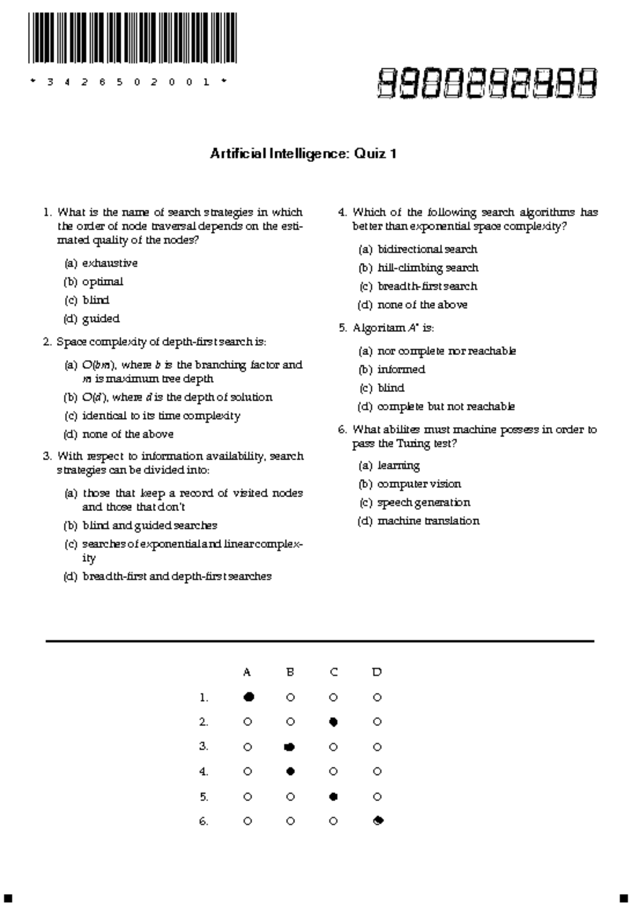
Fig. 1: Empty template of the seven-segment digit.

positions of the segments. Users need to colour-in the segments that form the desired digit using a pen or a felt-tip marker. An example of a properly filled-in test sheet is given in Fig. 2.

We define the problem of *number segmentation* as a problem of splitting the number image into a series of individual digit images. Each of the digit images is then processed using the devised classifier, thus solving the problem of number recognition.

It is not possible to predict the exact number and digit schemes that will be used for OCR. Authors of the forms should have the liberty to design their own arrangements of number fields in the document. Empty digit schemes that will be filled out by the end users can be created in several ways. For example, some authors might prefer to use the appropriate 7-segment font, while others might draw them manually. Therefore, the OCR system needs to be tolerant to dimension and shape variations of segments and digits. Additionally, the outline of the segments may be printed out in black or some nuance of grey.

While basic number patterns of segments to be filled are generally standardized, it should be noted that some ambiguities exist in the digit writing standards. There are (at least) two different glyphs for the digits “0”, “1”, “4”, “6”, “7” and “9”. For example, the digit “7”, written by filling the segments 1, 3 and 6, can be written with, or without the segment 2 filled (Fig. 1). Similar variations are possible for other digits mentioned above. In this stage of our work, these ambiguities are not taken into account. Instead, a strict digit writing convention is introduced to simplify the task of training the neural networks. We agree on segment filling standards as defined in the open-source 7-Segment font [6], as demonstrated in Fig. 3.



* 3 4 2 6 5 0 2 0 0 1 *

Artificial Intelligence: Quiz 1

1. What is the name of search strategies in which the order of node traversal depends on the estimated quality of the nodes?
   (a) exhaustive
   (b) optimal
   (c) blind
   (d) guided
2. Space complexity of depth-first search is:
   (a) $O(bm)$, where b is the branching factor and m is maximum tree depth
   (b) $O(d)$, where d is the depth of solution
   (c) identical to its time complexity
   (d) none of the above
3. With respect to information availability, search strategies can be divided into:
   (a) those that keep a record of visited nodes and those that don't
   (b) blind and guided searches
   (c) searches of exponential and linear complexity
   (d) breadth-first and depth-first searches
4. Which of the following search algorithms has better than exponential space complexity?
   (a) bidirectional search
   (b) hill-climbing search
   (c) breadth-first search
   (d) none of the above
5. Algoritam A^* is:
   (a) nor complete nor reachable
   (b) informed
   (c) blind
   (d) complete but not reachable
6. What abilites must machine possess in order to pass the Turing test?
   (a) learning
   (b) computer vision
   (c) speech generation
   (d) machine translation

	A	B	C	D
1.	●	○	○	○
2.	○	○	●	○
3.	○	●	○	○
4.	○	●	○	○
5.	○	○	●	○
6.	○	○	○	●

Fig. 2: An example of a filled-in test sheet with integrated answers sheet.

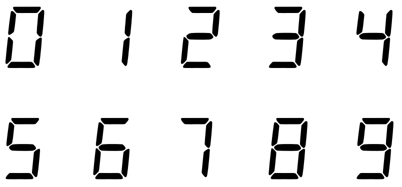


Fig. 3: Used seven-segment glyphs.

B. OCR from the user's point of view

The OCR system we are developing is intended for practical purposes. In this section, we briefly describe the recognition work-flow using the proposed system.

First, a template of the documents to be recognized needs to be provided to the system. The operator can use an empty copy of the form, or one that was already filled-in. After the scanning of the template is done, the operator needs to define the *form scheme.* A form scheme is a layout of *number fields*. It includes information about the position and size of each number field in the form sheet. Only one number per line is allowed. Numbers in multiple lines can be processed so that each line is defined as a separate number field.

After the form scheme is defined, the system performs the analysis of defined number fields: it identifies individual digits within the number fields, and queries the user if the correct number of digits is detected. If the analysis of a certain number field is not accurate, the user can manually define the number of the digits in that field. However, inaccurate detection of the number of digits in most cases means that the form template is corrupted, for example by poor scanning or filling. If no errors have been detected, the system can process all of subsequent forms of the defined template fully automatically.

C. Data Set Acquisition

Due to the fact that the recognition techniques we developed are based on neural networks, the appropriate sets for network training and validation needed to be gathered.

Our dataset was compiled by giving out sample forms, filled-in by volunteers. These forms were scanned and preprocessed using our segmentation system, in order to produce a data-set of accurately classified digits, filled by hand. After the preprocessing, the dataset was manually inspected, and poorly segmented digits, as well as the digits written in non-standard notation, were removed. The first version of our data set contains approximately 1.700 digits and is publicly available for download at [5].

III. Our Approach

In this work, we use various methods for digit recognition. All of these techniques are based on *multilayer feed-forward artificial neural network*. For each method, input images are grey-scale pictures of the document, taken in sufficiently high resolution. The appropriate neural network topologies are developed, using the help of a neural network topology optimizer based on the *genetic algorithm* (GA). To further improve the performance of the system, a neural networks committee is devised.

To facilitate the work-flow described in Sec. II.B, it is necessary to take several types of input pattern variations into account. The first type of the discrepancies is caused by the fact that form authors can use any type or size of the seven-segment font, as described in Sec. II.A. The second type is caused by minor anomalies that are likely to occur in the input images due to the limitations of the scanning technology. Slight translations or rotations of the input images, as well as a certain level of noise are likely to be introduced into the image even when using high quality scanners. Additionally, the colour of white paper is usually recognized as light grey instead of pure white. The system we are developing needs to be invariant to diversity of the input document styles, and the degradations that may occur during the scanning.

After the number field areas have been defined (as in Chap. 2.B), our system needs to know the exact positions of the digits within a number. Defining exact positions of digits could be done by the operator, but since clicking over dozens of digits is a tedious task, we wanted to provide an automated support for the digit segmentation. Our segmentation system searches for individual digits in each number field, and saves the number field parameters: number of digits, digit dimensions and positions into the *number field scheme*. This kind of analysis is demanding, therefore it is performed only for the template of the form.

The rest of the forms of the same scheme are scanned without further input from the operator. The positions of the digits inside a number field are known from the number field schemes. However, scanned images may result in pictures that are slightly translated or rotated when compared to the form scheme. The OCR system therefore needs to find the position of the entire number, and align the number field scheme with the image being recognized. The alignment process is much less demanding than the entire digit position detection.

The additional challenge was to give operators the possibility to use both empty and filled forms as templates. Filled-in forms may cause problems during segmentation, for instance if the user draws a line beyond segment limits, or smears the ink on the digits.

After the digits within the number fields have been isolated, they are recognized using our recognition techniques.

A. Recognition Techniques

We devised three digit recognition techniques. Each of them is characterised by different types of digit image features given to the network. As will be shown below, the methods that require more sophisticated preprocessing give smaller feature vectors, and yield better results, while those that employ preprocessing of lesser complexity provide feature vectors of large dimensions, as well as poorer results on our dataset.

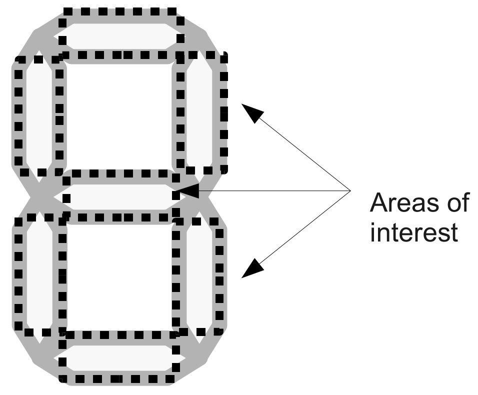


Fig 4: Interest areas method.

(1) The *interest areas method* is based on the digit segments' position detection (Fig. 4). An interest area is placed over each of the digit segments (dashed areas on Fig. 4). The feature vector is an array of seven real numbers in [0,1] interval. Each of the vector components corresponds to the average pixel value in each of the interest areas, i.e. segments, scaled to the [0,1] interval. The advantage of this method is small dimension of the input vector. However, the preprocessing required for this method is relatively complex.

(2) The *resolution reduction technique* utilizes input image re-sampling to a low resolution picture. In this approach, the feature vector consists of pixel values of the down-sampled image (Fig. 5). This method clearly increases the number of inputs to the network. We also get less reliable results with this method. However, its advantage is a significantly simpler preprocessing, as compared to other two of our methods.

When applying this method, it is necessary to make a trade-off between the detail level and input vector dimension. Higher resolution input vectors are able to provide more information about picture features, but the neural networks that work with such vectors must have a large number of inputs, thus making the training process highly demanding. On the other hand, lower resolutions provide much less detail, but the feature vectors have more reasonable dimensions. In our experiments, resolution of 10×20 pixels has given fair results.

(3) The *axis distribution analysis* method uses three distributions gathered from the digit image. The idea is to collect the average pixel values in n columns and m rows (Fig. 6). The first distribution is an array of average values of m rows along the y axis. The other two are the average

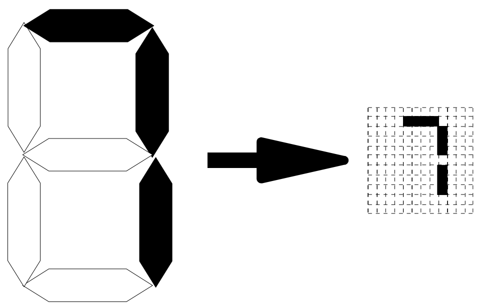


Fig. 5: Resolution reduction method.

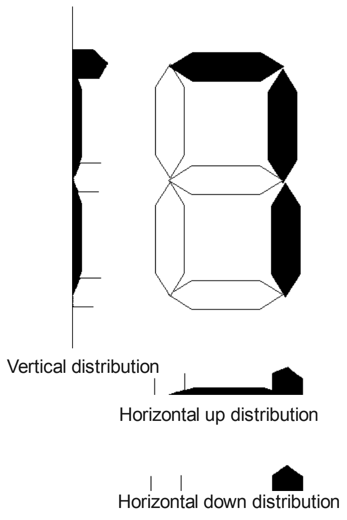


Fig. 6: Segmentation using average pixel value distributions.

pixel values in *n* columns along the *x* axis: one from the central segment and above, and one from the central segment below. Only one horizontal distribution is not sufficient, since digits “2” and “5” have identical distributions along the *x* and *y* axes.

For this method to work, a preprocessing, similar to the one performed in the interest areas method is necessary, in order to discover the placement of the central digit segment.

B. Neural Networks – an Evolutionary Approach

We use a multilayer feed-forward neural network as a classifier, trained for the task of digit recognition. It is trained on our dataset using back-propagation algorithm. The validation of each network is performed using *holdout validation* method, where test set size is 20% of our entire data-set. Neural network topology has a significant influence on the performance of the neural network training process, as well as on the classification and generalization efficiency. Therefore, we give special attention to the topology design issue.

All of the networks we use have ten outputs. Each of the network outputs corresponds to an output class i.e. a decimal digit. The network classifies the input sample into the class with the highest output value. The values on the outputs are in the interval [0, 1].

The neural network topology design is usually left to human experts, and often results in one, or few topologies considered optimal [1, 2, 3, 7]. No automatic way to systematically create optimal topology for a specific task is known [7]. The process of manual topology design usually consists of multiple trial-and-error iterations, led by intuition of the designer.

In this work, we perform experiments with both manual and an automated approach based on the GA to the optimal topology design issue.

A problem of determining the optimal neural network topology can be viewed as a combinatorial optimization problem of finding the best topology on a search space of all possible topologies. We consider GA appropriate for that purpose, due to its ability to reuse temporary results and its mechanisms that evade convergence to the local optima. Pseudo-algorithm of the topology optimizer we developed is given in Fig. 7.

```
Iterate {
    SelectThreeIndividuals();
    CrossTwoBetterAndReplaceWorst();
    MutateNewIndividual();
    TrainNewIndividual();
    EvaluateNewIndividual();
}
```

Fig. 7: Pseudocode of neural network topology optimizing algorithm.

Initial population for the genetic algorithm is generated randomly in order to scatter possible results throughout the solution space. Fitness value for each neural network is calculated, after training with sample set, based on its classification efficiency and the number of neurons in all layers. Topologies with greater efficiency, and less neurons in all layers have greater fitness.

Selection is random-based and any individual can be selected. Worst individual out of three is selected based on it’s fitness value, and is removed from the population. Single-point crossover of parent individuals, possibly mutated, results in a new individual, which may produce a slight performance improvement. Mutation operator adds or removes neurons or layers, and changes the activation threshold.

The process in most of the cases converges into several different network topologies, usually outperforming initial and trivial solutions.

C. Committee of Neural Networks

The performance of the neural networks we developed has shown to be from fair to very good, depending on the recognition technique and network topology used. Since multiple digit classifiers have been devised, the idea of combining their work into a more reliable one comes naturally.

In this stage of our work, we used *ensemble averaging*, as the simplest type of the committee design. The output of the committee is a weighted average of the network output vectors. The weight for each network output depends on the network reliability on the validation set.

IV. Preprocessing Methods

A. Number Segmentation

The segmentation system is needed for the digit isolation during the OCR, as well as test set acquisition. We frame the problem of segmentation as a problem of classifying the picture's pixels into two classes: digit areas and whitespace areas. After the areas of the picture have been classified into these classes, it is easy to locate digit edges and to cut-out pictures of separate digits.

The analysis needed for the segmentation is performed on the preprocessed version of the input image. The initial preprocessing converts the grey-scale image into a black/white image, using the average pixel value of the initial image as black/white threshold.

For each pixel row i and column j, we define the *row pixel sum* $s_r(i)$ and *column pixel sum* $s_c(j)$, as the sums of the pixel values in the i-th row and j-th column of the picture:

$$s_c(j) = \sum_{j=0}^{h} pix(i,j)$$

$$s_r(i) = \sum_{i=0}^{w} pix(i,j)$$

$pix(i, j)$ being the pixel value at the coordinate (i, j), and w, h being the row width and column height.

The segmentation is performed by analyzing the row/column pixel sums in the bitmap. An important assumption of our segmentation method is the existence of the *whitespace threshold* values of s_r and s_c, denoted s_{rt} and s_{ct}. The threshold values separate whitespace and digit rows/columns. For example, the columns having pixel sum below s_{ct} are columns of digit areas, and those having pixel sum above are columns of whitespace areas. With that in mind, the problem of classifying the pixel rows and columns into digit and whitespace can be restricted to finding the s_{rt} and s_{ct} values. Experiments have shown that the aforementioned assumption is true for almost all of the processed samples.

A sample of the x and y axis distributions of a typical number field is shown in Fig. 8. Let us consider the distribution of column pixel sums for this sample. The search for the threshold value s_{ct} is performed by the analysis of the histogram of s_c values. In the vast majority of scanned number fields, the s_c-histogram shows that the s_c values are dominantly grouped around two values: level of white, denoted s_{white} and the average s_c value of the digit areas, denoted s_{digit}. The threshold value s_{rt} is defined as the arithmetical mean of s_{white} and s_{digit} values.

The search for the s_{white} and s_{digit} values is tricky, since the histograms commonly include numerous local minima and maxima. The s_{white} and s_{digit} values are found by searching the local maximums of frequencies in the s_c-histogram that are at least d pixel values away from each-other, where d is an empirically determined constant.

After the value of s_{rt} has been found, determining the class of the pixel column i is performed using the formula:

$$class(i) = \begin{cases} digit\ area, & \text{for } s_c(i) \leq s_{ct} \\ white\ space\ area, & \text{for } s_c(i) > s_{ct} \end{cases}$$

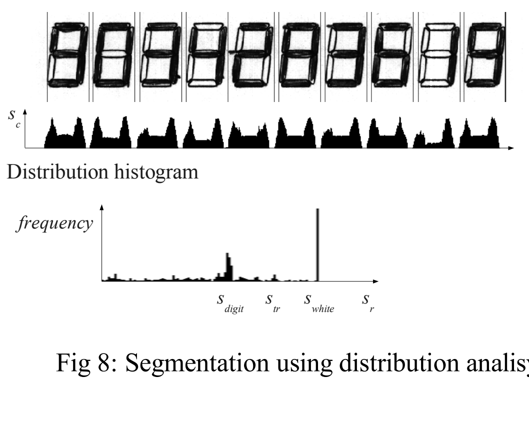


Fig 8: Segmentation using distribution analisys.

The task of the row classification is performed in an analogous manner. The column classification problem is more complex than rows classification, since it is very important to correctly detect each of the horizontal gaps between digits, while the task of the vertical analysis is only to determine the upper and lower limit of the y-axis interval in which the digits are placed.

After the entire x and y range of the picture has been split into digit and whitespace intervals, digit areas are determined as rectangular areas of pixels whose both coordinates are classified as digit area.

B. Digit Feature Extraction

Digit feature extraction is a preprocessing that converts an input image of a single digit into a vector of features that can be used as inputs for the neural network. Since we use three recognition techniques, a custom feature extraction needs to be devised for each of the methods described in Section II.A.

The *interest area method* uses the most complex feature extraction system. We assume that each digit is located in a frame that has the same dimensions for every digit of a single number (fixed-width digits). Additionally, we assume that relative positions of the segments in the frame are approximately equal. Each digit of the number field is separately analyzed to discover segment positions and sizes. The average parameters for all of them are calculated and assigned to the digit scheme.

The upper and lower edges of the digit are detected by analyzing s_r values of the digit image. The upper edge is placed on the first row with s_c lower than a predefined threshold value, when searching the value from the upper column to below. The lower is placed on the first row with s_c below the threshold value when searching from the bottom column up.

The problem of defining the vertical edges of the digit is more complex. Vertical border lines could not be assumed, since it is common to have the seven-segment digits italicized. Let us define $a_\varepsilon(i, j)$ as the arithmetic mean of the pixels in the ε-environment of the point (i, j). The ε-environment of the point (i, j) the set of pixels whose distance to the point (i, j) is less than ε. In each pixel row j, our algorithm searches for the minimum and maximum abscissa values $x_{min}(i)$, and $x_{max}(i)$ having the corresponding $a_\varepsilon(i, j) < a_{t\varepsilon}$.where $a_{t\varepsilon}$ is an empirically determined constant. The minimum and maximum abscissa values represent the left and the right border of the digit in a single pixel row. The left vertical borderline of the digit is found by calculating the linear regression line of the $x_{min}(i)$, and the right one is found by linear regression of $x_{max}(i)$ values. Inner digit edges are found in a similar manner. For the digits written in italic, a shear transformation that aligns them them into the digit frame with minimal size is used.

As a result of this analysis, our system is provided with a cleaned-up and properly placed picture, defined frame and segment areas. After the segment areas have been defined, it is trivial to calculate the average pixel values of each digit segment and bring them to the input of neural network.

The *resolution reduction technique* uses the simplest feature extraction mechanism. After being aligned, according to the digit scheme, digit pictures are down-sampled to a low resolution image using bicubic filtering, and then converted to a black/white image, using the average pixel value as the black/white threshold.

The feature extraction support for the *distribution analysis method* needs to be able to locate the central segment of the digit. This procedure is a subset of the preprocessing required for the the interest area method feature extraction. Therefore, the same procedure as in the interest area method is used. After the position of the central digit segment is detected, it is possible to calculate the average pixel values in the picture's rows, as well columns above and below the central segment.

V. RESULTS AND FUTURE WORK

The early tests of our system show highly encouraging results on the first version of the dataset. We analyzed the performance of neural networks with both manually and automatically designed topologies. The success rates of the automatically obtained ones are slightly higher than the manually designed ones. However, the manually designed topologies have significantly lower number of neurons and layers, thus being faster in their operation, and easier to train.

All of the individual neural networks have classification success rates higher than 90 percent. The committees of machines have shown a very high success rate, greater than 99%. In almost all of the experiments, a committee of machines achieved higher success rates than the individual networks. It has shown that the most reliable recognition technique is the interest areas method, achieving success rates of individual networks up to 99% on our validation set. The success rates of different recognition techniques and automatically generated neural network topologies are shown in Table 1. All of the considered networks are trained on the same set of samples, 80 percent in size of our data-set. The rest of the data-set was used for validation purposes.

TABLE 1:
THE PERFORMANCE OF AUTOMATICALLY DESIGNED NETWORK TOPOLOGIES

Topology	**Recognition method**	**Success rate (%)**
10	distribution analisys	96.61
25 65 26 65 10	distribution analisys	90.68
40 50 10	distribution analisys	90.96
20 40 20 10	distribution analisys	91.81
40 10	distribution analisys	98.02
7 7 10	interest areas	80.79
7 25 10	interest areas	97.74
18 10	interest areas	98.02
28 18 10	interest areas	99.15
14 18 21 10	interest areas	98.02
175 20 10	resolution reduction	97.74
210 10	resolution reduction	98.02
120 120 80 10	resolution reduction	98.87
30 50 10	resolution reduction	97.74
20 60 120 10	resolution reduction	97.74
Network Committee		99.15

TABLE 2:
THE PERFORMANCE OF MANUALLY DESIGNED NETWORK TOPOLOGIES

Topology	**Recognition method**	**Success rate (%)**
7 10	interest areas	98.02
10 10	resolution reduction	95.76
10 10	distribution analisys	97.46
Network Committee		98.59

We consider these results good enough to attempt a practical implementation of the system at our institution In our future work, we plan to improve many of the system components and additionally, to expand our data-set.

For example, we would like to enhance the segmentation reliability, by adding periodicity detection, and enhancing the number positioning algorithm. Another part of our plan is to implement the support for multiple digit glyphs for each digit, as described in Section II.A.

We would also like to develop warning report mechanism, that notifies the user in cases of low reliability results (for example, when dealing with poorly scanned or filled digits). Furthermore, we would like to test the performance of Hopfield neural networks, which seem suitable for the task we are facing [1] and investigate more advanced network committee approaches, such as boosting by filtering.

VI. CONCLUSION

As a result of this work, we created a base for a specific purpose OCR application. Aside from neural networks committee for recognition of seven-segment display digits, we developed a support for segmentation and feature extraction, necessary for our recognition methods. The results have shown to be highly promising, and in future work, we consider a practical application of the system at The Faculty of Electrical Engineering and Computing, University of Zagreb, Croatia.

REFERENCES

[1] D. J. C. MacKay, *Information Theory, Inference, and Learning Algorithms,* Cambridge University Press, 2003..

[2] M. Cheriet, N. Kharma, C. L. Liu, C. Y. Suen, *Character Recognition Systems,* John Wiley & Sons, 2007.

[3] R. M. Hristev, *The ANN Book*, 1st Edn., GNU Public Licence, 1998.

[4] S. V. Rice, F. R. Jenkins, T. A. Nartker, " The Fifth Annual Test of OCR Accuracy ", *Information Science Research Institute,* 1996.

[5] Data set of seven.segment display digits, available at: http://morgoth.zemris.fer.hr/neuraldigits

[6] Harvey Tyman's seven-segment font http://www.twyman.org.uk/Fonts/

[7] M. Matteucci, "ELeaRNT: Evolutionary Learning of Rich Neural Network Topologies ", Center for Automated Learning and Discovery , Technical Report N. CMU-CALD-02-103