An Integration of Online and Pseudo-Online Information for Cursive Word Recognition

Tal Steinherz, Ehud Rivlin, Nathan Intrator, and Predrag Neskovic

Abstract—In this paper, we present a novel method to extract stroke order independent information from online data. This information, which we term pseudo-online, conveys relevant information on the offline representation of the word. Based on this information, a combination of classification decisions from online and pseudo-online cursive word recognizers is performed to improve the recognition of online cursive words. One of the most valuable aspects of this approach with respect to similar methods that combine online and offline classifiers for word recognition is that the pseudo-online representation is similar to the online signal and, hence, word recognition is based on a single engine. Results demonstrate that the pseudo-online representation is useful as the combination of classifiers perform better than those based solely on pure online information.

Index Terms—Online, offline, handwriting, cursive, word recognition, classifier combination.

1 Introduction

Online cursive word recognition is a challenging task ([1], [2]). Currently, online recognition suffers from several weaknesses that involve sensitivity to stroke order, stroke number, and stroke characteristics variations ([3]). As similar shapes might be produced by different sets of strokes, two instances of the same letter or word may resemble each other in the image domain though they are associated with diverse online signals. This phenomenon is dominant at the beginning of words, after a pen-up pen-down motion and in the presence of letters that contain delayed strokes like *i* and *t*. Following the event of pen-up or, equivalently, at the beginning of a word, any writer may put the pen down in a different location and, hence, the next letter could be drawn in a unique manner. Delayed strokes may appear at any time and, consequentially, at any place, within the online signal. Fig. 1 presents two instances of the letter "o" that have the same topology—a single loop with no tails—and similar geometry—mostly concave with global extreme points at each side. Nevertheless, since each one of these letters was drawn from a different starting point (the beginning of stroke number 1), it resulted in a unique set of strokes. Another side of this phenomenon is observed in letters like *b*, *d*, *h*, *i*, and *l*, where an ascender that looks like a thick pole can be made by either one of an elongated loop or cusp form.

A special group of strokes that often cause inconsistency problems are the ones we term *redundant strokes*. There are many superfluous pixels that are byproducts of the continuous nature of cursive handwriting. On the one hand, such *redundant strokes* do not appear constantly because some writers use the pen-up pen-down option instead. On the other hand, in the time domain, adjacent strokes remain disjoint even if they meant to overlap, so it is only in the image domain that they coincide. See, for example, Fig. 2 where there is zoom-in on two instances of the letter "a." The difference between the two samples is the ligature that connects the letter in Fig. 2b to its preceding (strokes number 1 and 2). Another aspect of *redundant strokes* that is very common in real scenes is strokes that are repeated either intentionally or accidentally. Sometimes, the writer goes back and corrects himself. There are other times when he mistakenly produces a slip of "ink" at the end of a letter.

In a similar way, incomplete (open) loops are more frequent in the online representation. It seems that many writers count on stroke width and smearing effects to close the loops they sketch. Hence, it would be only in the real offline representation (a scanned bitmap image) that some loops mature. This phenomenon, which is demonstrated in Fig. 2a, is also considered a major problem in online recognition.

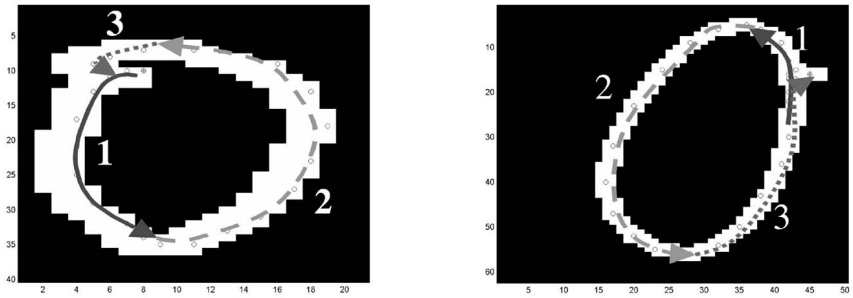
In this paper, we present a novel approach to extract an alternative list of strokes from an online signal. The resulting representation, which we term pseudo-online, is a two-dimensional signal of pixel locations along an imaginary uniformly distributed time axis—$(x(t), y(t))$.

Our aim is to provide a method that produces a new representation for a given word in which the above-mentioned phenomena will be minor. In order to achieve this goal, we suggest that one should refer to the associated bitmap image of the online word with the purpose of deriving the proposed alternative list of strokes. To generate this bitmap image, we simulate the image that would have been captured by scanning a real scene. The main emphasis is on giving strokes a significant width.

The main advantage of the bitmap image representation is the resemblance between instances of the same word. In this case, the deterministic algorithm that we describe next would derive a consistent list of strokes from the bitmap image of the word. In addition, many genuine strokes are filtered out because of the blurring effect and the lack of temporal information. This process reduces the number of redundant strokes. Using this approach, we propose a new strategy to

- *T. Steinherz and N. Intrator are with the Department of Computer Science, Tel-Aviv University, Ramat Aviv 69978, Israel. E-mail: {talstz, nin}@cs.tau.ac.il.*
- *E. Rivlin is with the Department of Computer Science, Technion, Haifa 32000, Israel. E-mail: ehudr@cs.technion.ac.il.*
- *P. Neskovic is with the Physics Department and Institute for Brain and Neural Systems, Brown University, Box 143, Providence, RI 02906. E-mail: pedja@brown.edu.*

Manuscript received 5 Feb. 2003; revised 27 Jan. 2004; accepted 7 Sept. 2004; published online 11 Mar. 2005.
Recommended for acceptance by V. Govindaraju.
For information on obtaining reprints of this article, please send e-mail to: tpami@computer.org, and reference IEEECS Log Number 118238.



(a) (b)

Fig. 1. Two instances of the letter "o" that were produced by diverse sets of strokes due to a different starting point, but resulted in similar shapes, i.e., the same topology—a single loop with no tails—and similar geometry—mostly concave with global extreme points at each side.

bootstrap an alternative list of strokes, the pseudo-online, from an online signal, using the bitmap image representation as a buffer. The idea is to regenerate an ordered list of strokes that would represent the input word in a different manner, mostly in view of spatial closeness between adjacent strokes. Thus, we produce new offline-like features that support an improved online recognition process.

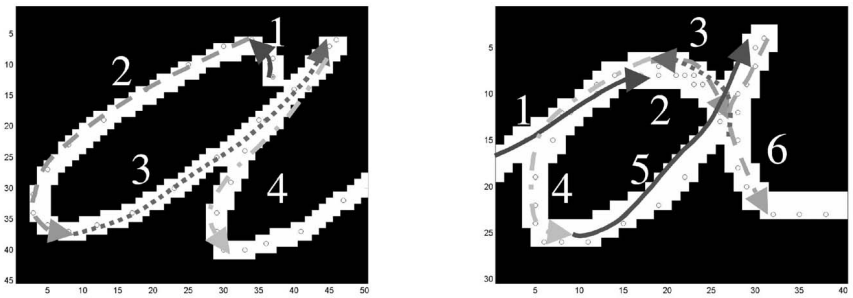
Several methods were proposed to do an offline to online transformation. They were usually referred to as *temporal information recovery* ([10], [19], [20]). Nonetheless, in contrast with these methods, our algorithm does not aim to trace the original path of the pen. This would have been less useful for the purpose of integration with the genuine online signal because there would have been no diversity between their outputs. Instead, we control the pseudo-online transformation to be complementary to the online signal and create true additional recognition critical information.

In order to serve the concept of a transformation that keeps resemblance between images, the information carried by the pseudo-online representation is analogous to the shapes of the characters creating the word. This is achieved by referring to the external contour of significant parts like ascenders, descenders, and loops. Furthermore, the image is always traversed from the leftmost pixel to the rightmost pixel and, in this way, a consistent order among the branching parts is enforced. For similar reasons, all connected components, with the exception of i-dots, are concatenated. The latter are virtually associated with the closest peak in the middle zone of the word.

The proposed pseudo-online representation enables one to recognize words that are not recognized easily given the genuine online signal. However, there are many other words for which recognition could be easier using the original signal alone. Usually, the pseudo-online representation would be inferior when valuable strokes disappear in the bitmap image. Sometimes, temporal information is also a must. This is the case, for example, when stroke order is the only way to distinguish between similar objects ([4], [5]).

In general, online classifiers are superior to pure pseudo-online classifiers. When one exploits offline-like features derived from bitmap images, he might lose valuable significant informative strokes that are filtered out ([4], [5]). In a similar way, offline classifiers are also limited to some extent ([6], [7], [8]).

As a result, one cannot replace the online representation with the pseudo-online. Nevertheless, in order to get closer to optimality, we suggest using them both. We propose working with a combination of classifiers trained individually on either online or pseudo-online representations, respectively. Thus, the heterogeneous classifiers would compensate each other. The transformation from online to pseudo-online via bitmap image should be configured to compensate and support the pure online representation, creating new features and, hence, new recognition ability. This will increase the group of recognizable words span by the combination of classifiers and maximize the overall recognition rate.



(a) (b)

Fig. 2. Two instances of the letter "a" that resemble each other in the image domain but present diverse sets of strokes. While the first sample starts with a pen-down operation, the second one is connected by a ligature to the preceding (strokes number 1 and 2). This ligature is hidden in the bitmap image because it was overlapped by the other strokes. The first sample also demonstrates a loop that was left open (strokes number 1 and 3 are not in touch), but became closed in the offline representation (in white).

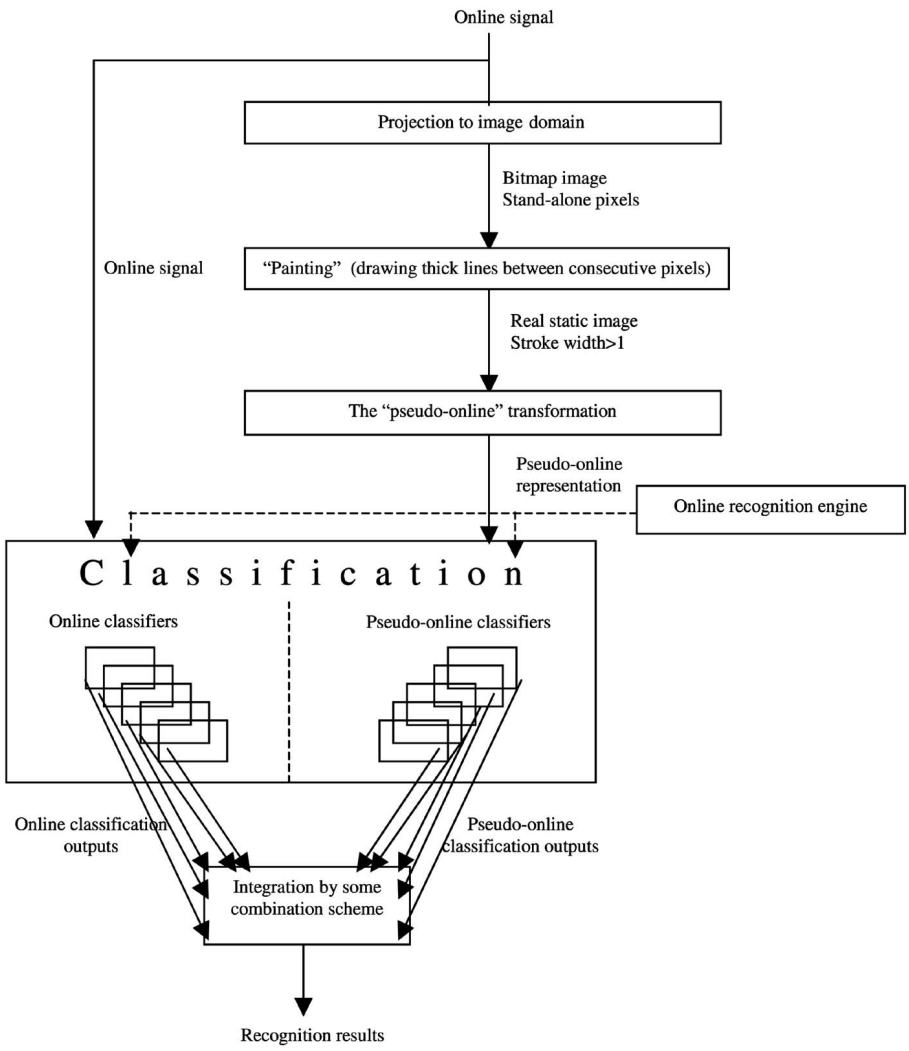


Fig. 3. Data flow in the proposed method. The online signal is duplicated and processed simultaneously by online and pseudo-online classifiers. The pseudo-online root presents some preprocessing stages that precede the recognition: projection to the image domain, drawing of thick lines between consecutive pixels, and the pseudo-online transformation. Both representations are then fed to several classifiers that are instances of the same online word recognition engine. The output results from both paths are collected and fused by a common combination scheme.

The fact that the pseudo-online is also in the form of pixels (strokes) as a function of (an imaginary) time is the key to a dual practice of a single online word recognition engine.

Therefore, one can utilize a single online word recognition engine to run both online and pseudo-online data. In this case, there will be two sets of trained instances of the same engine—for the genuine and pseudo-online signals, respectively. Both sets of classifiers are trained on the same data and the dissimilarity is gained from the unique extended preprocessing.

We propose runing online word recognition in two parallel paths—see Fig. 3. The given input word is duplicated and processed simultaneously by online and pseudo-online classifiers. In the pseudo-online root, a preprocessing stage precedes the recognition. It consists of three steps: First, the online signal is projected to the image domain, then consecutive pixels are connected by thick lines to establish meaningful stroke width, and, eventually, the pseudo-online signal is extracted. All the classifiers are instances of a single online word recognition engine. The outputs of all classifiers from both paths are collected and fused by a common combination scheme.

There were previous methods that suggested a combination of classifiers for online recognition ([9], [10]). Some of them proposed embedding offline-like data in the classical stroke representation during preprocessing ([11], [12], [13], [14]). Other methods suggested integration with pure offline word recognition systems at the postprocessing stage ([15], [16], [17], [18]). We propose using the same online word recognition engine to process both types of data. Thus, maintenance is simplified to a single engine with a single data set. Moreover, according to Velek et al. ([18]), who investigated online and offline classifier combinations, it requires some relation among the confidence values that are attached to candidates stemming from different classifiers in order to compare them. Sometimes, various normalization functions are necessary to overcome their inadequacies. Therefore, an integration of classifiers based on the same recognition engine is more straightforward.

Our experiments show that such a mixed combination of classifiers improves the recognition rate suggested by a combination of online classifiers solely. In general, online classifiers are not sensitive to the same words that the pseudo-online classifiers are. These results also demonstrate that the

Fig. 4. An instance of the word "complete" where the axis is colored in white and the various tarsi are colored in gray. Crossed points indicate intersections between the axis and meaningful tarsi where their dimension exceeds a threshold.

features conveyed by the pseudo-online representation are informative and useful.

The paper is organized as follows: Section 2 describes the proposed pseudo-online transformation. Section 3 explains the integration of online and pseudo-online classifiers. Section 4 presents experimental results that are analyzed and discussed in Section 5. We summarize with some insights and conclusions in Section 6.

2 The Pseudo-Online Transformation

In a series of papers in late 1980s and early 1990s ([21], [22]), Simon and Baret defined cursive handwriting as: "Displacing a pen from left to right in an oscillating movement, with loops, descendants (legs), and ascendants (poles)." Following this observation, the set of strokes that construct a word's image is divided as follows:

The main subset is an a posteriori concatenation of strokes that assemble the backbone, which is the shortest path from left to right, including loops on several occasions. We term this subset the axis. The other subsets are groups of connected strokes that produce branches which hang above (in case of ascenders) or below (in case of descenders) the axis. These subsets are called tarsi. Fig. 4 presents the separation of the word "complete" to axis, colored in white, tarsi, colored in gray, and to hanging points that are crossed.

Consequently, we propose a novel approach to transform a word's image to an ordered list of strokes ([23]): Follow the skeleton of the axis from the leftmost pixel until reaching the first intersection with a tarsus. Surround the tarsus by tracking its contour until returning back to the intersection point we started from. Continue along the axis to the next intersection with a tarsus and so on until the rightmost pixel is reached. Loops that are encountered along the axis are also surrounded completely.

In the next sections, we will elaborate on each module of the transformation. We start with the preliminary image generation procedure and the description of two necessary preprocessing steps: The concatenation of characters to provide a pure cursive word reflection and the extraction of a thin form for every loop in the word's image. Following is the method to compute the axis' skeleton, extract and order the tarsi and i-dots, plus postprocessing smoothing activities. Fig. 5 gives an overview of the process.

2.1 Preliminary Image Generation

Given $(x(t), y(t))$—a two-dimensional vector representing the genuine online signal and $up(t)$—an associated one-dimensional vector denoting pen-up motions, then the projected bitmap image is formed by initializing foreground pixels in the same locations captured by the digitizing device:

$$I_{i,j}=\begin{cases} 1 \text{ if } \exists t|(i,j)=(x(t),y(t)) \\ 0 \text{ otherwise.} \end{cases} \quad 1\le i\le|columns|, 1\le j\le|rows|$$

Next, thick lines are drawn between every pair of consecutive stand-alone pixels that share the same stroke. In this case, the line thickness would be determined by the desired stroke width that should be simulated. The resulting bitmap image is modified as follows:

$$I_{i,j}=$$
$$\begin{cases} 1 \text{ if } \exists t|(i,j) \text{ is on the thick line between} \\ \quad (x(t),y(t)) \text{ and } (x(t+1),y(t+1)) \text{ and } (up(t)=0) \quad 1\le i\le|columns|,1\le j\le|rows| \\ 0 \text{ otherwise.} \end{cases}$$

2.2 Connected Component Analysis

Our algorithm expects a truly cursive word of a single continuous pen movement. Unfortunately, there are many words that are not written as a single entity, so the word's image is built of several components. These words are preprocessed in order to merge the individual parts into a single element. First, the word's image is searched and the connected components are spatially ordered from left to right. Next, each pair of consecutive components is concatenated by an artificial "bridge"—a sequence of pixels between the two closest pixels across the neighboring components.

Nevertheless, if a connected component is heuristically identified as an i-dot, it is not bridged to any other component.

2.3 Loop Extraction

The objective of this step is to produce thin representations for all the loops in the word's image in order to serve the computation of the axis' skeleton that comes next.

An intuitive definition of a loop would be: a set of neighboring foreground pixels surrounding a "hole," i.e., a connected blocked group of background pixels in the word's image, where all foreground pixels are within *stroke width* distance from the "hole." We denote the thin representation of a loop as the minimal subset of these foreground pixels that form a closed outline. One group that satisfies this requirement is the complete collection of foreground pixels that have at least one 4-neighbor background pixel that belongs to the "hole." Fig. 6a illustrates a "hole" made of one background pixel. In general, the foreground pixels enumerated 14, 15, 16, 18, 19, 21, 22, and 23 are its 8-neighbors and pixels 15, 18, 19, and 22 are its 4-neighbors. Therefore, the last subset would be the thin representation of the loop—see Fig. 7a.

2.4 Computing the Axis' Skeleton

First, we create a graph $G(V, E)$ that is isomorphic to the word's image I. We define the following transformation:

Each foreground pixel $p_i \in I$ (belonging to the word's body) is associated with a unique node $v_i \in G$ in the graph. Therefore, there are N nodes, where N is the number of foreground pixels in the image. Next, all pairs of 8-neighbor foreground pixels $(p_i, p_j) \in I \times I$ are connected with an edge $e_k \in E$ between their associated nodes $(v_i, v_j) \in V \times V$. In this case, the neighboring matrix M is formed as follows:

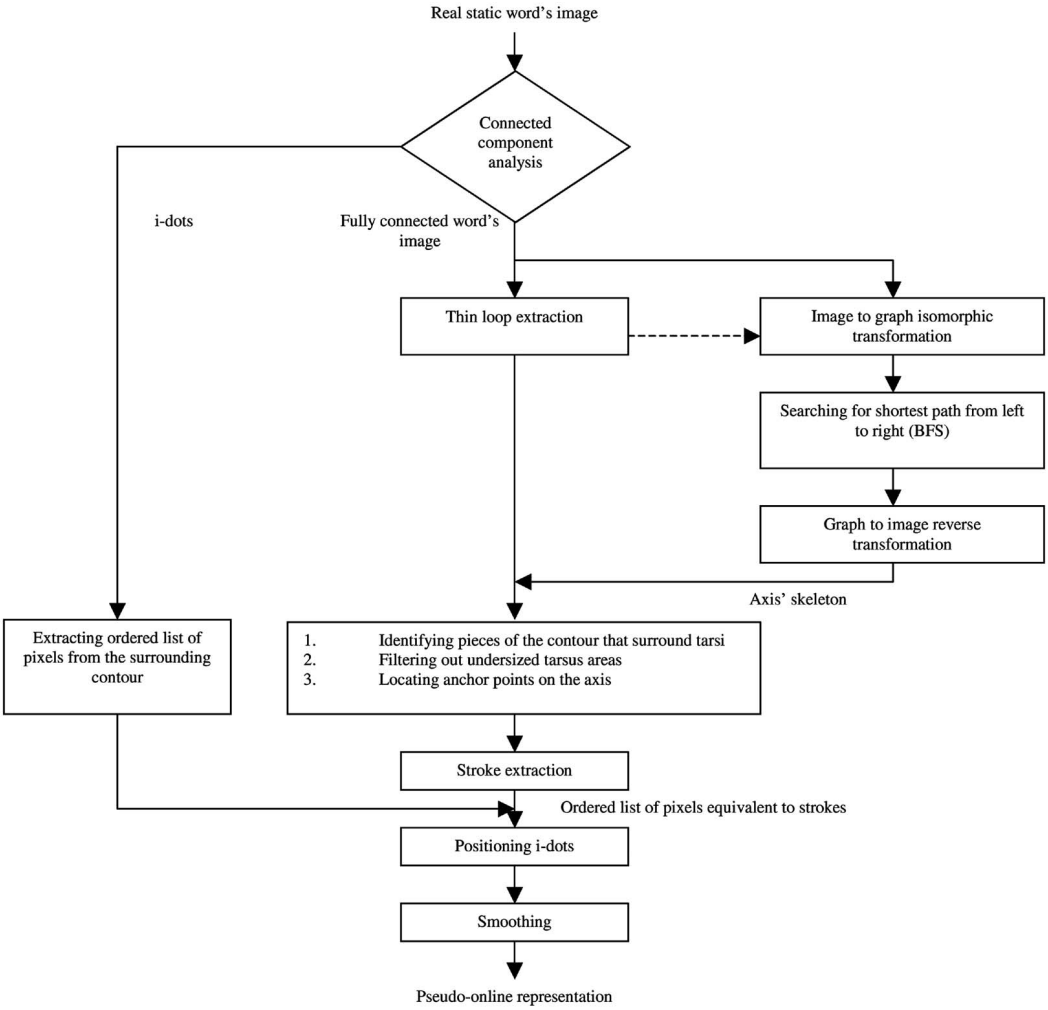


Fig. 5. Flow chart of the pseudo-online transformation. First, connected components are separated into main body and i-dots. Next, the axis' skeleton is computed using an isomorphic graph representation of the word's image and the output of a thin loop extraction procedure. Given the axis, the tarsi are extracted by identifying relevant pieces on the surrounding contour. The axis, tarsi, and i-dots in their new representation forms are merged and smoothed into an ordered list of pixels that are equivalent to strokes.

$$M_{i,j} = \begin{cases} 1 \text{ if } p_i \text{ and } p_j \text{ are 8-neighbors} \\ 0 \text{ otherwise} \end{cases} \quad 1 \le i, j \le N.$$

Fig. 6a shows an image of the word "co" where all body pixels are enumerated and, in Fig. 6b, there is an illustration of its isomorphic graph representation.

The next step is to group nodes that correspond to pixels of the same loop into a single special node. For this purpose, we use the thin representations of loops that were extracted in the previous step. Let $L = \{p_{l_1}, .., p_{l_m}\}$ be a set of pixels that form a loop in the word's image. Then, nodes $\{v_{l_1}, .., v_{l_m}\}$ are removed and node $\mathrm{v_{N+1}}$ is appended. All edges $\{e_{l_1}, .., e_{l_n}\}$ that connect pairs of nodes of the form $\{v_j, v_{l_i}\}$ are replaced with edges $\{e'_1, \ldots, e'_n\}$ that connect pairs of nodes of the form $\{v_j, v_{N+1}\}$, respectively. So, nodes that were connected to one of the deleted nodes are now connected to the new special node. The neighboring matrix is modified as follows:

$$\begin{aligned} M'_{i,j} &= M_{i,j} && i, j \notin \{l_1, \ldots, l_m\} \\ M'_{i,N+1} &= \begin{cases} 1 \text{ if } \exists \text{ k} \in \{l_1, \ldots, l_m\} \\ \quad |p_i \text{ and } p_k \text{ are 8-neighbors} \\ 0 \text{ otherwise} \end{cases} && 1 \le i \le N \\ M'_{N+1,i} &= M'_{i,N+1} && 1 \le i \le N. \end{aligned}$$

Fig. 7a shows in gray a group of pixels that form a loop. The resulting isomorphic graph after grouping their associated "loopy" nodes is shown in Fig. 7b.

At this stage, if p_1 is the leftmost pixel and p_N is the rightmost pixel, then the shortest path between v_1 and v_N is isomorphic to the axis' skeleton of the word's image. A standard Breadth First Search (BFS) algorithm is used to find the shortest path in the graph. After this stage, we completed the computation of the axis' skeleton.

Fig. 8a shows an illustration of the shortest path in the graph representation and Fig. 8b shows the associated axis' skeleton in the word's image.

2.5 Processing the Tarsi

Defining the contour to be the background pixels on the edge of the word's body, there are three tasks to be carried out with regards to the tarsi. First, pieces of the contour that surround tarsus areas should be identified. Next, undersized pieces that are presumed noise are filtered out. Finally, we locate the optimal points along the axis' skeleton to which the remaining tarsi's contours will be connected. In what follows, we describe these tasks in details.

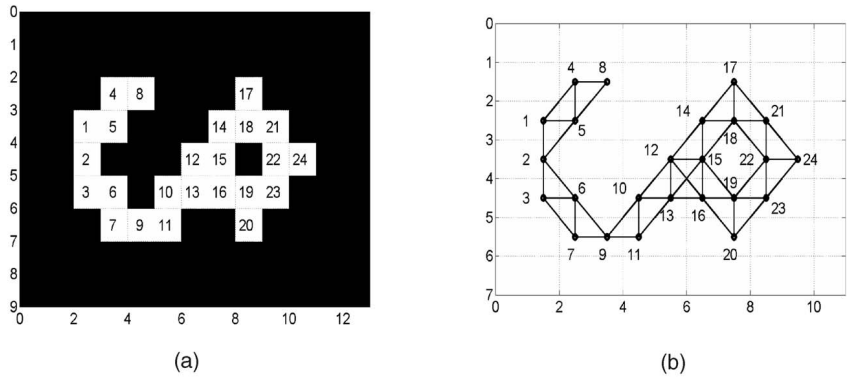


Fig. 6. A demonstration of the isomorphic graph that represents the image of the word "co" in (a) and (b), respectively. Each foreground pixel (belonging to the word's body) is associated with a unique node in the graph. All pairs of nodes that represent 8-neighbor foreground pixels are connected with an edge.

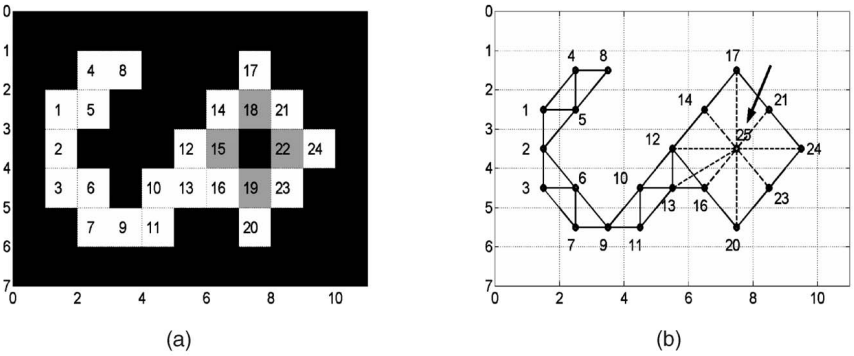


Fig. 7. The image of the word "co" with gray-colored pixels forming the thin representation of the loop in the letter "o" and the isomorphic graph following the grouping of all the nodes that were associated with these pixels to a single special node pointed by the arrow—in (a) and (b), respectively. The new special node is connected to each one of the nodes that were connected to the nodes it replaced.

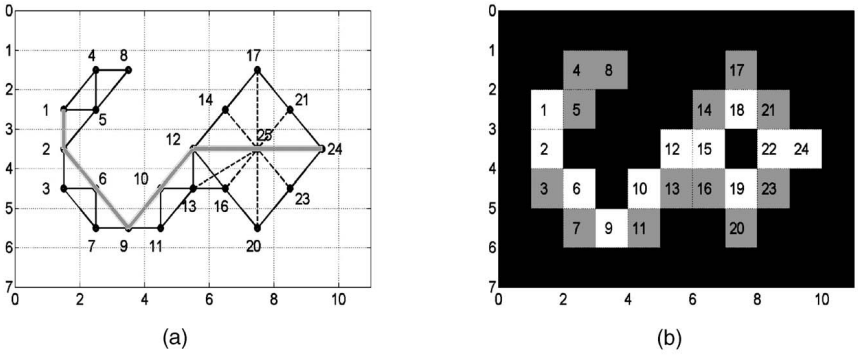


Fig. 8. (a) The light gray-marked shortest path between nodes 1 and 24 that are associated with the leftmost and rightmost pixels, respectively. (b) The white-colored pixels in the word's image that are represented by the nodes on the shortest path. This group includes the four pixels of the loop (15, 18, 19, and 22) that were represented by a single special node (25). All of these pixels form the axis' skeleton of the word's image.

In order to identify tarsus areas, we calculate the *Geodesic* distance between the axis' skeleton and each one of the pixels on the contour. The *Geodesic* distance is the minimum number of body pixels that separate two body points. Fig. 9a shows the output of this procedure on the contour of the word "co" with its computed axis' skeleton.

Next, we use the *Geodesic* distance to separate the pixels on the contour into two groups of axis-based and tarsus-oriented, respectively.

Axis-based pixels include those pixels that are up to a threshold away from the axis' skeleton. Usually, one would set the threshold according to the averaged stroke width so that the associated pixels would relate to the axis' contour. In this particular case, the strokes gained their width by an artificial painting procedure and the expected averaged stroke width is known in advance. Pixels whose distance from the axis' skeleton exceeds the threshold identify the tarsi.

Each subset of consecutive pixels on the contour forms a separate tarsus. To complete the second task, tarsi whose size is less than a predefined number of pixels are considered to be noise. Fig. 9b shows the three tarsi that were extracted from the image of the word "co": a true tarsus and two undersized tarsi (crossed).

The last task is to locate the hanging points along the axis' skeleton to which the remaining tarsi's contours will be connected. To achieve that each tarsus is heuristically matched with one pixel on the axis' skeleton. The tarsus'

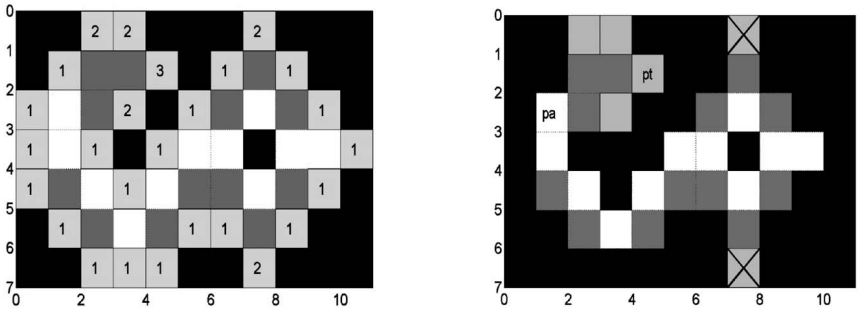


Fig. 9. (a) The *Geodesic* distance between the axis' skeleton (colored in white) and each one of the pixels on the contour (colored in light gray). (b) Pixels whose distance from the axis' skeleton exceeds the threshold of 1. These pixels identify pieces of the contour that surround tarsus areas (undersized tarsi are crossed). The extremist pixel on the tarsus' contour and its closest pixel on the axis' skeleton are marked as pt and pa, respectively.

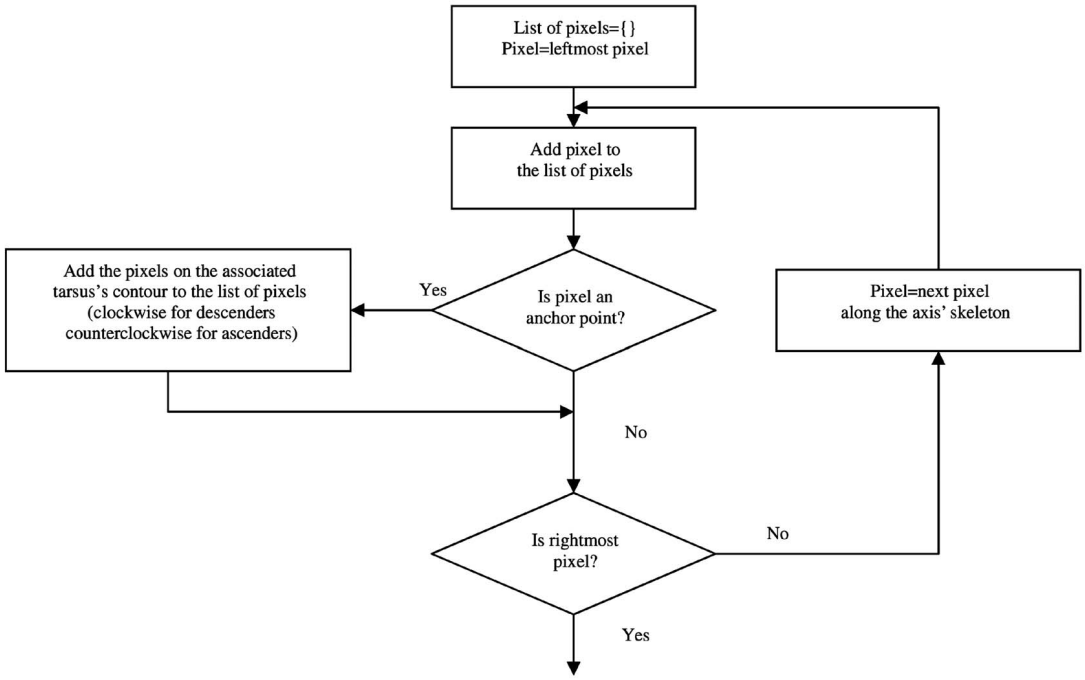


Fig. 10. Stroke extraction flow chart starting on the leftmost pixel and following the skeleton of the axis until reaching the rightmost pixel. Upon meeting an anchor point that denotes an intersection with a tarsus, the pixels on the tarsus' contour are added to the ordered list of pixels.

contour will be connected to this pixel in the stroke extraction phase. Let pt be the extremist pixel on the tarsus' contour (see Fig. 9b), then we heuristically select its closest pixel pa on the axis' skeleton (see Fig. 9b) as the connecting pixel which we term anchor point.

2.6 Stroke Extraction—Getting an Ordered List of Pixels

Given an axis' skeleton with several anchor points and a set of associated tarsi represented by continuous pieces of the contour, we use the following scheme to extract an ordered list of strokes from the processed image of the word (see Fig. 10 with the flow chart and Fig. 11a for a full demonstration of the ordered pixels):

1. *pixel = leftmost pixel* (the pixel indexed 1 in Fig. 11a).
2. If *pixel is an anchor point*, then
   a. If *anchored tarsus is an ascender*, then surround contour from *anchor point* → *last pixel on tarsus' contour* → *first pixel on tarsus' contour* → back to *anchor point* in a counterclockwise manner (pixels indexed 2-7 in Fig. 11a).
   b. If *anchored tarsus is a descender*, then surround contour from *anchor point* → *first pixel on tarsus' contour* → *last pixel on tarsus' contour* → back to *anchor point* in a clockwise manner.
3. *pixel = next pixel along the axis' skeleton.*
4. While *pixel <> rightmost pixel* (pixel indexed 21 in Fig. 11a) goto Step 2.

Note that loops on the axis' skeleton are surrounded counterclockwise from the *left intersection point with the axis* ("entrance" painted in light gray and indexed 14/18 in Fig. 11a) → *right intersection point with the axis* ("exit" painted in light gray and indexed 16/20 in Fig. 11a) → back to the *left intersection point with the axis* → back to the *right intersection point with the axis* (pixels indexed 14-20 in Fig. 11a). This definition is necessary in order to preserve full connectivity and continuity without pen-up pen-down motions. We term

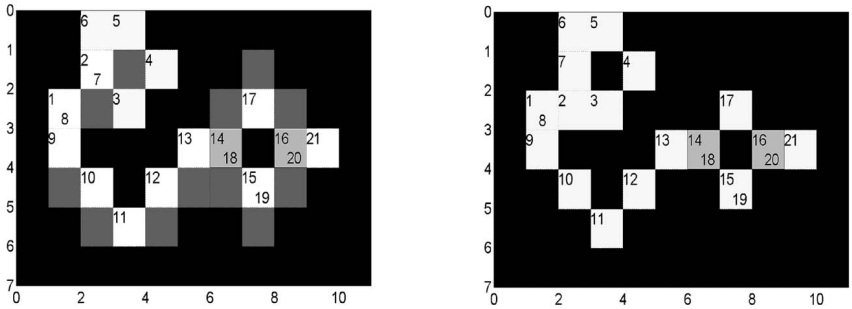


Fig. 11. The resulting ordered list of pixels (colored in white and light gray and indexed 1 to 21) that were extracted from the image of the word "co," before and after smoothing—in (a) and (b), respectively. The light gray-colored pixels represent the "entrance" and "exit" of the loop, respectively. This ordered list of pixels equivalent to strokes is the pseudo-online representation of the word.

the resulting list of strokes that includes the axis and the tarsi the main body.

2.7 Handling i-Dots

If the word's image contains a stand-alone i-dot, it does not appear in the ordered list of strokes that was extracted so far as it was not bridged to the main connectivity component that was handled in the previous stages. We consider i-dots that are separated from the main body of the word as a special kind of tarsi. Ordered lists of pixels extracted by surrounding their contours would represent these components. Each list should be embedded in the appropriate position within the ordered list of strokes that has been extracted previously (Sections 2.4, 2.5, and 2.6). We chose to position the i-dots immediately following the closest local maxima found in the main body. By so doing, a pen-up pen-down motion is created. Fig. 12 demonstrates the positioning of the i-dot representation in the pseudo-online ordered list of pixels that was extracted from the word "in."

2.8 Postprocessing

It turns out that the resulting strokes may contain small fluctuations (see, for example, the sequence of the first three pixels, indexed 1-3, in Fig. 11a). Therefore, we utilize a second order smoothing procedure to correct vertical fluctuations of single pixels.

Let $(x(t), y(t))$ be the ordered list of pixels that create the pseudo-online representation of a given word, then the new signal is smoothed as follows:

$$y'(t) = \begin{cases} y(t-1) \text{ if } (y(t-1) = y(t+1)) \\ \quad \text{and } ((x(t), y(t-1)) \in I) \\ y(t) \text{ otherwise} \end{cases} \qquad 2 \le t \le T-1.$$

Fig. 11b shows the smoothed ordered list of pixels.

2.9 The Pseudo-Online Representation

Going back to the examples that were shown in Section 1—Figs. 1 and 2, we hereby present the pseudo-online representations that were extracted from these letters—Figs. 13 and 14, respectively. The pseudo-online transformation generated an equivalent list of strokes for the two instances of the "o" and "a," respectively. The key to the consistency is the fact that letters are always connected to the left either originally or artificially and, therefore, the trajectory starts on one of the leftmost pixels. From there on, the similarity between the pseudo-online signals is achieved by the deterministic flow of the transformation algorithm—see the charts in Figs. 5 and 10. In this case, the resemblance between loops and tarsi of two letters (words) are translated to similarity in the pseudo-online representations because both loops and tarsi are always traversed isomorphic to their shape.

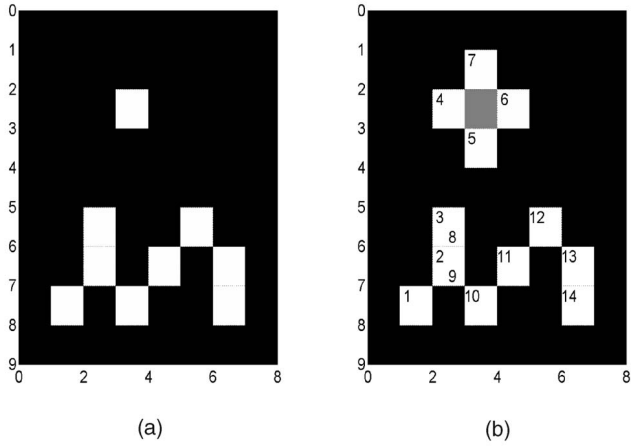


Fig. 12. An ordered list of pixels that gives the pseudo-online representation of the word "in"—(a) and (b), respectively. The i-dot is represented by an ordered sublist of pixels that surround it. This sublist is positioned immediately following the closest local maxima found in the main body and it is separated by a pen-up pen-down motion.

3 An Integration of Online and Pseudo-Online Classifiers

The pseudo-online representation is not a substitute for the genuine online signal. Nevertheless, it may contribute to improving the recognition rate of an online word recognition system. In order to achieve this goal, one should integrate two subsets of classifiers trained on online and pseudo-online signals, respectively. Thus, recognition is performed by a mixed combination of classifiers—see Fig. 3.

In order to have a reliable integration, multiple classifiers of each kind—online and pseudo-online, are required. Otherwise, it may be difficult to make a decision when the two classifiers disagree.

The fact that the pseudo-online is also given in the form of pixels (strokes) as a function of (an imaginary) time enables one to practice a single online word recognition

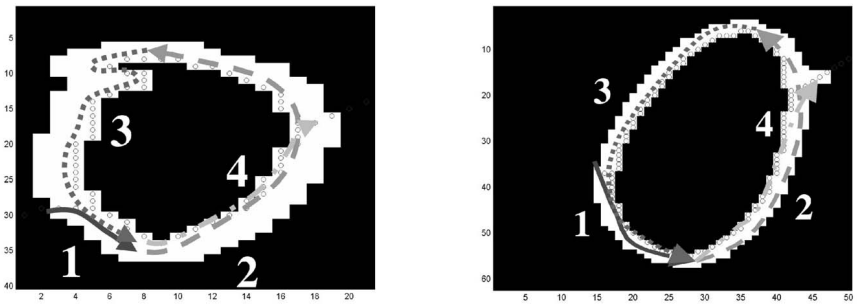


Fig. 13. Similar pseudo-online representations that were extracted for the two instances of the letter "o" that were originally produced by diverse sets of strokes (see Fig. 1). The consistency is achieved because the pseudo-online transformation always searches a letter from the leftmost pixel to the rightmost one in a deterministic way.

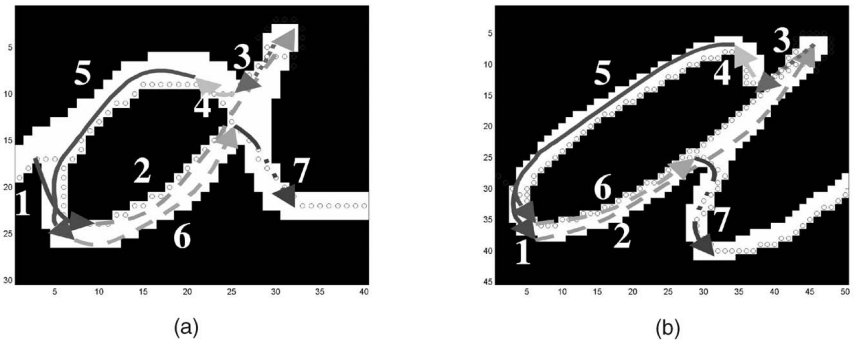


Fig. 14. Similar pseudo-online representations that were extracted for the two instances of the letter "a" that were originally produced by diverse sets of strokes (see Fig. 2). The consistency is achieved because the pseudo-online transformation always searches a letter from the leftmost pixel to the rightmost one in a deterministic way.

engine to learn both the online and pseudo-online representations. In this case, the online classifiers would be the result of training instances of the recognition engine with different parameters on the genuine online signal. In a similar way, the pseudo-online classifiers are consequences of training the same instances of the recognition engine on the pseudo-online representation of the same data set.

There are many advantages to a situation where all the classifiers in a combination rely on the same recognition engine and are trained on the same data set. It simplifies maintenance throughout the whole process: training, testing, and usage. Furthermore, the integration is more straightforward.

The proposed recognition strategy is not limited to a specific online word recognition engine. The classification module refers to the recognition engine as a black box. It can be used with any kind of recognition engine that satisfies the following requirements: *Trainability* in the sense of the ability to learn patterns of original signals that are given in the form of online signals and *Versatility*, which means that there are certain preset parameters that enable training of several different classifiers where, each one of them is more sensitive to different words.

Once there are two subsets of classifiers for online and pseudo-online, respectively, a combination scheme to fuse their outputs in order to create an ensemble is required. We used the following combination schemes investigated by Kittler et al. ([16]), one at a time:

- **Majority vote**: The chosen class would be the one that received the most votes from the mixed set of classifiers.
- **Max rule**: The chosen class would be the one with the highest confidence among all the optional classes suggested by the mixed set of classifiers.
- **Sum rule**: The chosen class would be the one that gained the highest accumulated confidence after summing up all confidences suggested for the same class by different classifiers.

When the majority vote is used, it often happens that two different classes win the same number of votes. Ties of this kind can be resolved by either one of the max rule or the sum rule.

4 Experimental Results

This section demonstrates that a combination of online and pseudo-online classifiers is superior to a combination of online classifiers solely. The comparison shows that, for subsets of online classifiers, there exist mixed subsets of the same size with online and pseudo-online classifiers that achieve higher recognition rates.

We conducted two series of experiments where, in each one of them, we have trained a set of online classifiers and a matching set of pseudo-online classifiers. In the first series, there were six classifiers of each kind. In the second series, the number of classifiers was doubled to give a total of 12 classifiers of each kind.

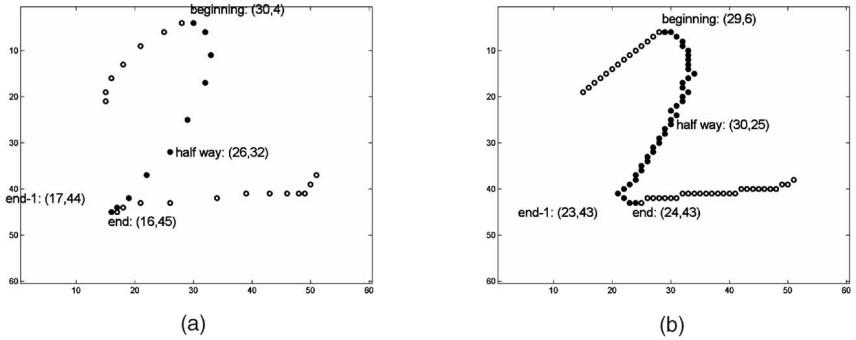


Fig. 15. An illustration of the main stroke derived from the online and pseudo-online representations of the digit "2." The local maxima and minima determine the beginning and end of the stroke. The feature vectors that characterize these strokes are: $(16-30=-14, 45-4=41, 26-30=-4, 16-17=-1, 1:1)$ and $(24-29=-5, 43-6=37, 30-29=1, 24-23=1, 1:1)$. (a) represents the online strokes and (b) represents the strokes obtained by the pseudo-online transformation.

In each series, the results of the first experiment were confirmed by a second experiment where we have retrained and retested the same classifiers on different training and test sets, respectively. The sizes of the data sets that were used in the second experiment were about two thirds of the ones used in the first experiment. All test sets that were utilized throughout the series were disjoint.

In each experiment of the second series, we used the same training, validation, and test sets that were used in the corresponding experiment of the first series. The second series was meant to test the changes in the contribution of the pseudo-online classifiers when new instances of online classifiers are introduced and the total number of classifiers of each kind increases.

The pseudo-online classifiers were trained, validated, and tested on the transformed data sets that were used by the online classifiers according to the methodology explained in Section 3 and the flow chart that was presented in Section 1 (see Fig. 3).

We used the three combination schemes that were discussed before: majority vote, max rule, and sum rule. We have also computed an upper bound to the combined recognition rate—denoted the "or rule." A combination scheme can only recognize an input word that was recognized correctly by at least one of the classifiers. We define the *upper bound recognition rate* that can be achieved by a subset of classifiers as the percentage of input words that were recognized correctly by at least one classifier. Let $TP_i(j)$ be a binary variable indicating whether classifier i correctly recognized the input word j and let W_n be the number of input words, then the *upper bound recognition rate* for a combination of classifiers $i_1, \ldots, i_n$ is

$$\mathrm{TP}r_{i_1,\ldots,i_n} = \sum_{j=1}^{W_n} (\mathrm{TP}_{i_1}(j) | \ldots | \mathrm{TP}_{i_n}(j)) \Big/ W_n.$$

In general, significant improvement over a combination of online classifiers solely was achieved when some of these classifiers were replaced with pseudo-online classifiers and a mixed combination of the same size was formed.

4.1 The Online Word Recognition Engine

Unfortunately, there are not many online word recognition engines that are public domain and satisfy both the trainability and versatility requirements. We chose the engine developed by Neskovic et al. ([24], [25]). This engine is inspired by properties of human vision and reading strategy and is based on a Time Delay (Space Displacement) Neural Network ([26], [27]). It enables one to train various classifiers that are distinguished from one another by the size of the receptive field of each model. The receptive field is analogous to the neighborhood of a letter in a cursive word where it has mutual interaction with other letters.

Neskovic et al.'s engine divides the input signal into strokes that comply with Hollerbach's observations ([28]). Each stroke is a line between two consecutive points with zero velocity in the vertical y direction, i.e., between a local minima (maxima) point and the local maxima (minima) point that follows it. All strokes are copied to feature vectors that are fed to the recognition engine as a stream.

Each stroke is characterized by five features ([29]): the x and y lengths of the stroke, the net displacement of the pen in the x direction halfway through the stroke, the velocity in the x direction at the end of the stroke, and the ratio between the spatial frequency in the x direction w_x and the spatial frequency in the y direction w_y.

The x and y lengths of a stroke are the sizes of its projection on the x and y axes, respectively. Halfway is determined by time, i.e., if a stroke starts at t_1 and ends at t_2, the halfway would be the point at $t = t_1 + (t_2 - t_1)/2$. Spatial frequency is a parameter that measures the amount of oscillations, i.e., the proportion between the total and net motion in the direction. The ratio is given in three quantum values (2:1, 1:1, or 1:2).

Since the pseudo-online signal does not convey real quantitative temporal information (the time interval between two consecutive pixels is constant and the absolute distance between them is always either 1 or $\sqrt{2}$), the velocity along each axis (x or y) has only qualitative meaning—steady state (0) or direction $(-1, +1)$. Fig. 15 illustrates the transformation of the data representation into feature vectors by the recognition engine. It demonstrates the use of extreme points to segment the feature vectors both in the online and pseudo-online representations and gives some insight to the type of features used by the engine.

In comparison with other online engines, the one we have selected presents a poor feature vector. However, the fact that it involves only basic stroke-related features enables us to evaluate the net contribution of the pseudo-online representation as an alternative ordered list of strokes.

4.2 The Data Set

We have tested our system with a data set of online cursive words compiled by Rumelhart ([29]). This is an extension of the *HP* data set that can be found in the UNIPEN collection

TABLE 1
A Comparison between the Recognition Rates of Pure Online and a Mixture of Online and Pseudo-Online Combinations of Six Classifiers Trained on a Data Set of 68 Writers × 862 Words and Tested on a Data Set of 18 Writers × 862 Words under Various Combination Schemes

Combination	Majority Vote	Max Rule	Sum Rule	Or Rule
All 6 online classifiers	71.2	70.8	71.3	78.5
4 online and 2 pseudo-online classifiers (on top of the experimented combinations)	75.0	73.9	75.1	84.4
Difference	3.8	3.1	3.8	5.9

TABLE 2
Mean Percent Recognition Rates and Standard Deviations of Six Pure Combinations of Six Online Classifiers and Six Mixed Combinations of Four Online and Two Pseudo-Online Classifiers

Combination	Majority Vote	Max Rule	Sum Rule	Or Rule
All 6 online classifiers	73.1±2.4	72.7±2.4	73.1±2.4	80.2±1.7
4 online and 2 pseudo-online classifiers (on top of the experimented combinations)	76.3±2.8	75.2±2.7	76.5±2.9	85.3±2.4
Difference	3.2±1.6	2.5±1.4	3.4±1.7	5.1±2.4
P-value	<0.01	<0.01	<0.01	<0.01

All sets of online classifiers were trained on different data sets of 46 writers × 862 words and tested on disjoint data sets of 11 writers × 862 words. All subsets of pseudo-online classifiers were trained and tested on the pseudo-online representations of the same data sets. The P-value provides the significance confidence level of the improvement of the mixed combinations over pure online combinations. The student t test was used to calculate the significance.

([30]). The data set provides one sample for every word per writer. The collection of words creates a lexicon of 862 words. A raw data file representing an input word contains information about x and y pen positions recorded every 10 milliseconds.

From the online data set, we generated its pseudo-online clone using the transformation described in Section 2. The bitmap image of a word was produced by a linear concatenation of all neighboring pixels between every pair of pen-down pen-up operations. Each line was then thickened in order to gain the real stroke width of three pixels at the average.

4.3 First Series of Experiments

The bases for the first series of experiments were six instances of the recognition engine that were compiled with different parameters to simulate different sizes of letter neighborhoods.

Experiment 1. For the first experiment, we used a training set of 68 writers and a test set of another independent 18 writers. Since each writer produced 862 words, there were 58,616 words in the training set and 15,516 words in the test set. A subset of the test set—9 of the 18 writers—was used as a validation set. This group of writers was introduced throughout the training phase for the purpose of calculating the variability in the expected error. The validation set was used for deciding when to stop the training process.

Each instance of the recognition engine was duplicated and trained separately on the online and pseudo-online representations of the training set. Thus, we have trained six online and six matching pseudo-online classifiers with mean percent recognition rates and standard deviations of 33.9 ± 14.8 and 31.2 ± 4.3, respectively.

Table 1 presents the percent recognition rates for the combination of all six online classifiers in comparison with a mixed combination of four online and two pseudo-online classifiers that achieved the highest recognition rate of all mixed combinations that were tested. The recognition rates refer to the various combination schemes that were mentioned above. The sum rule showed an improvement of 3.8 percent. The *upper bound recognition rate* was improved by 5.9 percent.

Experiment 2. The next experiment was meant to confirm the significance of the results that were achieved in Experiment 1. In order to do so, we have repeated the training and tests done in Experiment 1 another six times with different cross-validation training and test sets, respectively.

The data set that was used for these trials consisted of 66 out of the 68 writers that were used for training in Experiment 1. The same lexicon of 862 words was utilized for each writer. The group of 66 writers was randomly divided into six disjoint test sets of 11 writers. Each test set was introduced to classifiers that were trained on 46 of the remaining writers. In this case, we used the last nine writers for the on-training validation process that was explained above. In each trial, we have duplicated and retrained the six instances of the recognition engine on the online and pseudo-online representations of the appropriate training set, respectively. In this way, we have reproduced six online and six pseudo-online classifiers for every trial.

Table 2 lists the percent mean recognition rates and standard deviations of the six pure combinations of the six online classifiers and the six mixed combinations of the four online and two pseudo-online classifiers under the various combination schemes. The two pseudo-online classifiers to be used in the mixed combination were chosen from the performance of Experiment 1, where a disjoint test set of 18 writers was used. The sum rule provided the best combination results of 3.4 percent improvement. The upper bound on combination performance gave 5.1 percent improvement. The *student t test* was used to study the significance of the improvement in performance when combining pseudo-online together with online classifiers. The reported P-values indicate that the improvement under all combination schemes was significant at $P < 0.01$ confidence level.

TABLE 3
A Comparison between the Recognition Rates of Pure Online and a Mixture of Online and Pseudo-Online Combinations of 12 Classifiers Trained on a Data Set of 68 Writers × 862 Words and Tested on a Data Set of 18 Writers × 862 Words under Various Combination Schemes

Combination	Majority Vote	Max Rule	Sum Rule	Or Rule
All 12 online classifiers	77.7	76.3	77.8	88.5
7 online and 5 pseudo-online classifiers (maximized the *upper bound recognition rate*)	80.1	76.9	80.4	92.1
Difference	2.4	0.6	2.6	3.6

TABLE 4
Mean Percent Recognition Rates and Standard Deviations of Six Pure Combinations of 12 Online Classifiers and Six Mixed Combinations of Seven Online and Five Pseudo-Online Classifiers

Combination	Majority Vote	Max Rule	Sum Rule	Or Rule
All 12 online classifiers	79.7±3.1	78.5±3.2	79.8±3.1	89.4±1.9
7 online and 5 pseudo-online classifiers (maximized the *upper bound recognition rate*)	81.9±2.8	79.3±3.0	82.3±3.0	92.9±2.0
Difference	2.2±0.8	0.8±0.6	2.5±0.7	3.5±0.9
P-value	<0.01	<0.02	<0.01	<0.01

All sets of online classifiers were trained on different data sets of 46 writers × 862 words and tested on disjoint data sets of 11 writers × 862 words. All subsets of pseudo-online classifiers were trained and tested on the pseudo-online representations of the same data sets. P-values were calculated as in Table 2.

4.4 Second Series of Experiments

The second series of experiments extends the first series. In this case, we tested the progress of the improvement in the recognition rate of a mixed combination of classifiers when the number of classifiers increases. In addition to the six instances of the recognition engine that were used in the first series, we compiled another six instances to simulate new sizes of letter neighborhoods.

Experiment 3. Using the same training set of 68 writers from Experiment 1, we have separately trained a copy of each one of the six additional instances of the recognition engine on its online and pseudo-online representations, respectively. Thus, given the classifiers that participated in Experiment 1, both the online and pseudo-online sets of classifiers were extended to the size of 12 members with mean percent recognition rates and standard deviations of 33.4 ± 14.2 and 30.6 ± 3.6, respectively. Next, the additional classifiers were also tested on the same test set of 18 writers that was given to the original ones during Experiment 1.

Then, the pure combination of the 12 online classifiers was compared with a mixed combination of online and pseudo-online classifiers of the same size that maximized the *upper bound recognition rate* among all possible 12-classifier combinations. In this case, the subset of classifiers $l_1, \ldots, l_{12}$ maximizes the *upper bound recognition rate* among all combinations of 12 classifiers if

$$l_1, \ldots, l_{12} = \arg \max_{i_1,\ldots,i_{12}} (TPr_{i_1,\ldots,i_{12}}) \qquad 1 \leq i_1, \ldots, i_{12} \leq 2 * 12.$$

There were seven online and five pseudo-online classifiers in the subset that was found in this way. While this choice may not be optimal, it suffices to demonstrate significant improvement.

Table 3 presents the percent recognition rates for the combination of all 12 online classifiers in comparison with the mixed combination of the seven online and five pseudo-online classifiers that maximized the *upper bound recognition rate*. The results refer to the various combination schemes that were used in the first series. In this case, the sum rule showed an improvement of 2.6 percent and the *upper bound recognition rate* was improved by 3.6 percent.

Experiment 4. In a similar way to the one we described in the first series of experiments, we have repeated the training and tests done in Experiment 3 with the six cross-validation data sets. In each trial, we have retrained the 12 instances of the recognition engine on the online representation of the same training set of 46 writers that was used in the corresponding trial of the first series. In addition, we used the pseudo-online representation of that training set to retrain the five instances that were required to produce the five pseudo-online classifiers of the mixed combination that was selected. In this way, we have produced 12 online and five pseudo-online classifiers for every trial.

In each trial, we tested both the resulting pure combination of the 12 online classifiers and the resulting mixed combination of the seven online and five pseudo-online classifiers on the same test set of 11 writers that was used in the corresponding trial of the first series.

Table 4 lists the percent mean recognition rates and standard deviations of the six pure combinations of the 12 online classifiers and the six mixed combinations of the seven online and five pseudo-online classifiers under the various combination schemes. The sum rule provided the best combination results of 2.5 percent improvement. The upper bound on combination performance gave 3.5 percent improvement. The reported P-values indicate that the improvement under the majority vote and sum rule combination schemes continues to be significant at $P < 0.01$ confidence level. A similar confidence was obtained for the improvement in the *upper bound recognition rate* on combination performance. The max rule showed lower improvement and significance compared with the sum rule and majority vote.

Experiment 5. In the last experiment, we have tested and compared all possible mixed combinations of the N-best online classifiers and M-best pseudo-online classifiers $(0 \leq N, M \leq 12)$. The latter are considered fair approximations for the optimal mixed combinations of $N + M$ classifiers. The setup of Experiment 3 was used to train the individual classifiers and test the various

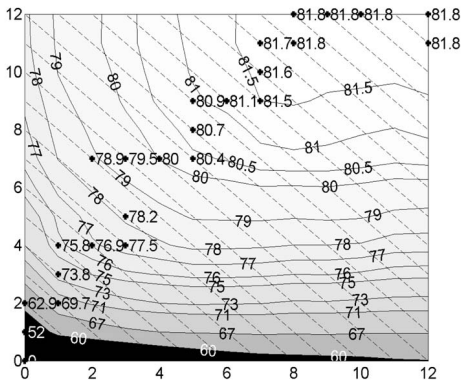


Fig. 16. A contour plot that illustrates recognition rates achieved by numerous mixed combinations of the N-best online classifiers—y-axis and M-best pseudo-online classifiers—x-axis $(0 \leq N, M \leq 12)$. The dashed diagonal lines represent subsets of combinations with the same number of classifiers. The "+"s mark the top ranked combination in every subset and the recognition rate it attained.

combinations. In this case, only the leading sum rule was implemented. Fig. 16 provides a contour plot graphical illustration of the 169 recognition rates that were achieved. Each dashed diagonal line presents a subset of combinations with the same number of classifiers. A "+" marks the top ranked combination in every subset and the recognition rate it attained. In the mixed combinations of 8-13 classifiers, we witnessed an improvement of 1.4-2.6 percent over combinations of the same size with maximum online classifiers. In this case, there were 2-5 pseudo-online classifiers in the top ranked mixed combinations.

5 Discussion

The resolution of the dissimilarities between the online and pseudo-online signals is at the letter level. There was some evidence that pseudo-online classifiers prefer words with complicated letters like *f* and *k* or words that begin with the letters *a*,*b*, *h*, *i*, and *l*. However, in general, the association between the input words that were better recognized by the pseudo-online classifiers and specific letters was not conclusive. This is likely due to the fact that the recognition engine averages the confidences of all letters in a word and masks local peaks. Nevertheless patterns of improvement could be seen at the word as well as the writer levels. The collection of results obtained in Experiment 4 shows that there were 110 word classes (12.8 percent) in which at least seven word samples (10.6 percent) were correctly recognized only by the combination with the pseudo-online classifiers. The ordered histogram presented in Fig. 17a summarizes, for each word class, the percentage of word samples that were correctly recognized only by the mixed combination of classifiers.

At the writer level, we observed that there were 12 writers (18.2 percent) for whom at least 65 of the words they produced (7.5 percent) were correctly recognized only by the combination with the pseudo-online classifiers. More than a 12 percent difference was obtained for the top two writers. The ordered histogram that summarizes the percent differences for all the writers is given in Fig. 17b.

Following the analysis of the mixed combination performance, we did some investigation to learn about the correlation among the pseudo-online classifiers in comparison with the correlation between online and pseudo-online classifiers, respectively. Let O_i and PO_i be the number online and pseudo-online classifiers that correctly recognized the ith input word, respectively, then the histogram of all possible positive values of $PO_i - O_i([1, \ldots, 12])$ that were observed in the 18-writer test set $(1 \leq i \leq 18 * 862 = 15,516)$ is given in Fig. 18a. Similarly, the distribution of the input words for which $PO_i \geq n (1 \leq n \leq 12)$ and $O_i = 0$ is shown in Fig. 18b. The statistics demonstrate that the pseudo-online classifiers share unique characteristics that reflect on a subdomain of recognizable input words. For example, there were 909 input words (5.9 percent) that were correctly recognized by at least one pseudo-online classifier and neither one of the 12 online classifiers.

Fig. 19 presents two of the leading words in this group that were recognized by at least 10 of the 12 pseudo-online classifiers.

To complete the picture, we have computed the mean diversity of the errors between a pair of online, a pair of pseudo-online and an online plus a pseudo-online pair of classifiers, respectively. Let $Cl_i(j)$ be the word class to which the input word j was classifieds by classifier i, then the diversity of the errors made by classifiers i_1, i_2 is given by:

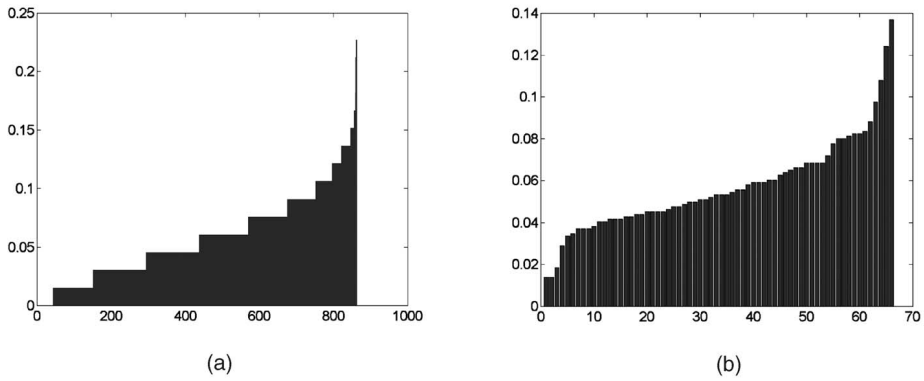


Fig. 17. (a) Percentage of word samples per word class that were correctly recognized by a mixed combination of seven online and five pseudo-online classifiers and were not recognized by a combination of all 12 online classifiers. The histogram summarizes recognition results on a data set of 66 writers $\times$ 862 word classes. (b) Percentage of word samples per each writer that were correctly recognized by a mixed combination of seven online and five pseudo-online classifiers and were not recognized by a combination of all 12 online classifiers. The histogram summarizes recognition results on a data set of 66 writers $\times$ 862 word classes.

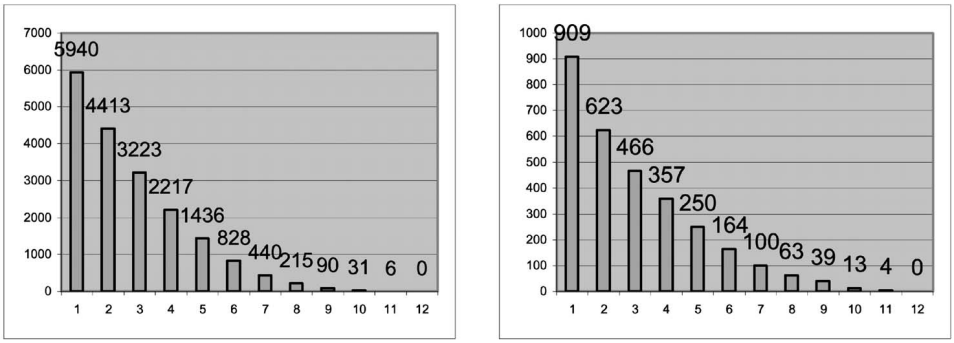


Fig. 18. (a) Number of input words that were recognized by at least n more pseudo-online classifiers than online classifiers $(1 \leq n \leq 12)$. The histogram summarizes recognition results on a data set of 18 writers $\times$ 862 word classes. (b) Number of input words that were recognized by at least n pseudo-online classifiers and neither one of the online classifiers $(1 \leq n \leq 12)$ for the above-mentioned data set.

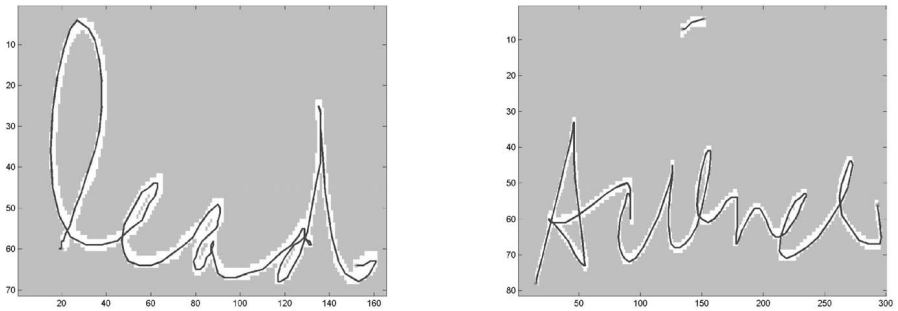


Fig. 19. Two samples of input words that were recognized correctly by at least 10 of the 12 pseudo-online classifiers and neither one of the 12 online classifiers. The word "lead" in (a) presents an open loop at the "a" and a ligature that interferes with the loop of the "d." Both problems were overcome in the bitmap image and the pseudo-online representation that was derived from it. The word "science" in (b) presents a delayed i-dot that is not positioned within the main body and a "c" that resembles an "e." In the latter case, this form of "c" is more likely to be recognized by pseudo-online classifiers because of the tarsus representation that was demonstrated in Section 2.

$$DIV_{i_1,i_2} = \frac{\sum_{j=1}^{W_n} (Cl_{i_1}(j) = Cl_{i_2}(j) | \mathrm{TP}_{i_1}(j) = 0)}{\sum_{j=1}^{W_n} (Cl_{i_1}(j) = Cl_{i_2}(j))}.$$

The results show correspondence among the various online and pseudo-online classifiers—$86.4\% \pm 6.1$ and $80.0\% \pm 5.9$, respectively. This is the antithesis of the high diversity that was found between pairs of online and pseudo-online classifiers—$94.3\% \pm 3.3$.

6 Conclusions

The recognition rates that were achieved by the mixed combinations of online and pseudo-online classifiers improve previous results that were accomplished on this data set ([24], [25], [29]). We have demonstrated that the pseudo-online representation is useful and provides additional information that can be easily translated into significant recognition rate improvement. We have shown that the variability in the recognition rate improvement is reduced when more classifiers (with different letter neighborhood sizes) are used. In the latter case, the number of pseudo-online classifiers that were part of the optimal combination was increased, emphasizing the important contribution of the pseudo-online representation to the genuine online signal. Moreover, it proves that the improvement was orthogonal to the improvement that could be obtained by introducing only new online classifiers.

This is further supported by the fact that the significance of the improvement was not changed when combining 12 classifiers, while the absolute recognition rate has jumped from 76.5 percent to 82.3 percent (in the case of the sum rule combination scheme). The *upper bound recognition rate*, which indicates the best performance that could be achieved from the collection of classifiers at hand (while not indicating a specific strategy for achieving such combination), retains the significance in the improvement while indicating far better absolute recognition results. For example, in the case of 12 classifiers, the *upper bound recognition rate* is over 92.9 percent compared to 85.3 percent in the case of 6-classifier combination. As an optimal combination scheme is outside the scope of this paper, this result is sufficient to indicate that the pseudo-online representation does add information that cannot be obtained by optimizing a combination of online classifiers only.

References

[1] T. Wakahara, H. Murase, and K. Odaka, "On-Line Handwriting Recognition," *Proc. IEEE*, vol. 80, no. 7, pp. 1181-1194, 1992.

[2] C.C. Tappert, C.Y. Suen, and T. Wakahara, "The State-of-the-Art in On-Line Handwriting Recognition—A Survey," *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 12, no. 8, pp. 787-808, Aug. 1990.

[3] G. Seni, R.K. Srihari, and N. Nasrabadi, "Large Vocabulary Recognition of On-Line Handwritten Cursive Words," *IEEE Trans. Pattern Analysis and Machine Intelligence,* vol. 18, no. 7, pp. 757-762, 1996.
[4] R. Plamondon and S.N. Srihari, "On-Line and Off-Line Handwriting Recognition: A Comprehensive Survey," *IEEE Trans. Pattern Analysis and Machine Intelligence,* vol. 22, no. 1, pp. 63-84, Jan. 2000.
[5] R. Seiler, M. Schenkel, and F. Eggimann, "Off-Line Cursive Handwriting Recognition Compared with On-Line Recognition," *Proc. Int'l Conf. Pattern Recognition,* pp. 505-509, 1996.
[6] A. Vinciarelli, "A Survey on Off-Line Cursive Word Recognition," *Pattern Recognition,* vol. 35, no. 7, pp. 1433-1446, 2002.
[7] T. Steinherz, E. Rivlin, and N. Intrator, "Off-Line Cursive Script Word Recognition—A Survey," *Int'l J. Document Analysis and Recognition,* vol. 2, nos. 2-3, pp. 90-110, 1999.
[8] S.N. Srihari, "High Performance Reading Machines," *Proc. IEEE,* vol. 80, no. 7, pp. 1120-1132, 1992.
[9] H. Nishida, "An Approach to Integration of Off-Line and On-Line Recognition of Handwriting," *Pattern Recognition Letters,* vol. 16, no. 11, pp. 1213-1219, 1995.
[10] S. Jaeger, "On the Complexity of Cognition," *Proc. Int'l Workshop Frontiers in Handwriting Recognition,* pp. 291-302, 2000.
[11] M. Hamanaka, K. Yamada, and J. Tsukumo, "On-Line Japanese Character Recognition Experiments by an Off-Line Method Based on Normalization-Cooperated Feature Extraction," *Proc. Int'l Conf. Document Analysis and Recognition,* pp. 204-207, 1993.
[12] M. Okamoto, A. Nakamura, and K. Yamamoto, "Direction Change Features of Imaginary Strokes for On-Line Handwriting Character Recognition," *Proc. Int'l Conf. Pattern Recognition,* pp. 1747-1751, 1998.
[13] S. Jaeger, S. Manke, J. Reichert, and A. Waibel, "On-Line Handwriting Recognition: The Npen++ Recognizer," *Int'l J. Document Analysis and Recognition,* vol. 3, no. 3, pp. 169-180, 2001.
[14] S. Manke, M. Finke, and A Waibel, "Combining Bitmaps with Dynamic Writing Information for On-Line Handwriting Recognition," *Proc. Int'l Conf. Pattern Recognition,* pp. 596-598, 1994.
[15] H. Kang, K. Kim, and J. Kim, "A Framework for Probabilistic Combination of Multiple Classifiers at an Abstract Level," *Eng. Applications of Artificial Intelligence,* vol. 10, no. 4, pp. 379-385, 1997.
[16] J. Kittler, M. Hatef, R. Duin, and J. Matas, "On Combining Classifiers," *IEEE Trans. Pattern Analysis and Machine Intelligence,* vol. 20, no. 3, pp. 222-239, Mar. 1998.
[17] H. Tanaka, K. Nakajima, K. Ishigaki, K. Akiyama, and M. Nakagawa, "Hybrid Pen-Input Character Recognition System Based on Integration of On-Line-Off-Line Recognition," *Proc. Int'l Conf. Document Analysis and Recognition,* pp. 209-212, 1999.
[18] O. Velek, S. Jaeger, and M. Nakagawa, "A New Warping Technique for Normalizing Likelihood of Multiple Classifiers and its Effectiveness in Combined On-Line/Off-Line Japanese Character Recognition," *Proc. Int'l Workshop Frontiers in Handwriting Recognition,* pp. 177-182, 2002.
[19] D.S. Doermann and A. Rosenfeld, "Recovery of Temporal Information from Static Images of Handwriting," *Int'l J. Computer Vision,* vol. 15, nos. 1-2, pp. 143-164, 1995.
[20] G. Boccignone, A. Chianese, L.P. Cordella, and A. Marcelli, "Recovering Dynamic Information from Static Handwriting. Pattern Recognition," vol. 26, no. 3, pp. 409-418, 1993.
[21] J.C. Simon, "Off-Line Cursive Word Recognition," *Proc. IEEE,* vol. 80, no. 7, pp. 1150-1161, 1992.
[22] J.C. Simon and O. Baret, "Regularities and Singularities in Line Pictures," *Int'l J. Pattern Recognition and Artificial Intelligence,* vol. 5, no. 1-2, pp. 57-77, 1991.
[23] T. Steinherz, N. Intrator, and E. Rivlin, "A Special Skeletonization Algorithm for Cursive Words," *Proc. Int'l Workshop Frontiers in Handwriting Recognition,* pp. 529-534, 2000.
[24] P. Neskovic and L. Cooper, "Neural Network-Based Context Driven Recognition of On-Line Cursive Script," *Proc. Int'l Workshop Frontiers in Handwriting Recognition,* pp. 352-362, 2000.
[25] P. Neskovic, P. Davis, and L. Cooper, "Interactive Parts Model: An Application to Recognition of On-Line Cursive Script," *Proc. Advances in Neural Information Processing Systems,* pp. 974-980, 2000.
[26] M. Schenkel, I. Guyon, and D. Henderson, "On-Line Cursive Script Recognition Using Time Delay Neural Networks and Hidden Markov Models," *Machine Vision and Applications,* vol. 8, no. 4, pp. 215-223, 1995.
[27] O. Matan, C.J.C. Burges, Y.L. Cun, and J.S. Denker, "Multi-Digit Recognition Using a Space Displacement Neural Network," *Proc. Advances in Neural Information Processing Systems,* pp. 488-495, 1992.
[28] J. Hollerbach, "An Oscillation Theory of Handwriting," *Biological Cybernetics,* vol. 39, pp. 139-156, 1981.
[29] D.E. Rumelhart, "Theory to Practice: A Case Study—Recognizing Cursive Handwriting," *Computational Learning and Cognition: Proc. Third NEC Research Symp.,* pp. 177-196, 1993.
[30] I. Guyon, L. Schomaker, R. Plamondon, M. Liberman, and S. Janet, "Unipen Project of On-Line Data Exchange and Recognizer Benchmarks," *Proc. Int'l Conf. Pattern Recognition,* pp. 29-33, 1994.

Tal Steinherz received the BSc degree in mathematics, physics, and computer science from the Hebrew University in Jerusalem. Currently, he is a PhD student in the Computer Science Department at Tel-Aviv University. His research interests are online and offline cursive word processing and recognition, document analysis, and writer model.

Ehud Rivlin received the BSc and MSc degrees in computer science and the MBA degree from the Hebrew University in Jerusalem and the PhD from the University of Maryland. Currently, he is an associate professor in the Computer Science Department at the Technion, Israel Institute of Technology. His current research interests are in machine vision and robot navigation.

Nathan Intrator received the PhD degree in applied mathematics from Brown University in 1991. He is an associate professor (research) at the Institute for Brain and Neural Systems at Brown University and an associate professor in the Department of Computer Science at Tel-Aviv University. Professor Intrator's research focuses on statistical and machine learning methods for data fusion from multiple experts. Recently, he has been studying biosonar and extended his results to data fusion to fusion of multiple echo returns for echo localization. His earlier work included object representation and recognition and dimensionality reduction via exploratory projection pursuit methods. He developed various methods for estimating predictor confidence which can be easily used for sensor fusion as well as classification and detection based on multiscale representations. He has been on the organizing committee of the Neural Information Processing Systems (NIPS) since 1991 and has co-organized a number of workshops at those conferences. He was on the organizing committee of the multiple classifier systems conference for the last five years and organized a NATO summer school on this topic in 2002.

Predrag Neskovic received the BSc degree in theoretical physics from Belgrade University, Yugoslavia, in 1990 and the PhD degree in physics from Brown University in 1999. He is currently an assistant professor (research) at the Institute for Brain and Neural Systems at Brown University. His research focuses on developing biologically inspired models for object recognition and visual perception. In particular, he is interested in constructing recognition systems that utilize contextual information during the recognition process. His research interests are mainly within the fields of pattern recognition and computer vision. He holds a US patent on an invention related to recognition and segmentation of cursive script.