Recognition of Handwritten Numerical Fields in a Large Single-Writer Historical Collection

Marius Bulacu Axel Brink Tijn van der Zant Lambert Schomaker
Artificial Intelligence Institute, University of Groningen, The Netherlands
(bulacu, a.a.brink, tijn, schomaker)@ai.rug.nl

Abstract

This paper presents a segmentation-based handwriting recognizer and the performance that it achieves on the numerical fields extracted from a large single-writer historical collection. Our recognizer has the particularity that it uses morphing during training: random elastic deformations are applied to fabricate synthetic training character patterns yielding an improved final recognition performance. Two different digit recognizers are evaluated, a multilayer perceptron (MLP) and radial basis function network (RBF), by plugging them into the same left-to-right Viterbi search framework with a tree organization of the recognition lexicon. We also compare with the performance obtained when no dictionary is used to constrain the recognition results.

1. Introduction

Archives around the world are in possession of kilometers of bookshelves with valuable historical documents. A very large portion is handwritten, as machine-printing was not always as convenient and widespread as today. Gaining electronic access to this wealth of information is only possible by building Google-like search engines. The challenge relies in converting the scanned document images into a searchable representation. Handwriting recognition has therefore the potential of becoming the cornerstone determining the success of such ambitious undertakings. In response to a query, the users are offered access to the scanned images to be able to fully research the original documents and also directly appreciate their graphical beauty. However, handwriting recognition will be the driving technology enabling the search and retrieval process.

Handwritten word recognition is dominated by the segmentation-free generative approach using hidden Markov models (HMMs). The alternative approach, followed also in this paper, uses a split-and-merge strategy: possible segmentation points are generated in excess and the correct subset is chosen by a search process based on the recognition scores produced by a discriminative character recognizer [4, 5, 6, 8]. This segmentation-based approach can capitalize on the experience accumulated in isolated character recognition.

An established method to improve the performance of pattern recognition systems is to use synthetic training data [2, 7, 9]. In this paper, we investigate the utility of artificial data in building a segmentation-based handwriting recognizer. The tests are carried out on the numerical fields extracted from a large single-writer historical collection.

2. The KdK dataset

Our research is performed on the archive of Kabinet der Koningin (KdK) - the Cabinet of the Dutch Queen [1]. This collection is in the custody of *Nationaal Archief* in The Hague and covers almost 200 years (1798-1988). It is very relevant historically because it holds a record of all important forms of government intervention in the Netherlands (laws, decisions, official appointments). All these state decisions are described by a short summary paragraph (see Fig. 1). Large portions of the KdK are written by a single writer (the clerk in office). Our goal is to build a search engine to allow users to find and retrieve only the paragraphs with the relevant content from the entire collection.

The KdK documents have an administrative nature and numerical fields are heavily used for identifying and referencing the official decisions. Recognizing these numerical fields is an important step, because they can be effectively used as hyperlinks, providing a good entry to the KdK collection. Each KdK paragraph is identified by the year (1903), month, day and decision number. Page references are used in the header / footer of the KdK documents. Layout analysis [3] allows the accurate extraction of three types of numerical fields which will be used in this paper: days, decision numbers and page references (see Fig. 1). The other numbers dispersed throughout the text require a separate approach as they must be recognized together with the content of the paragraphs. The three types of numerical fields were extracted from the complete document images, deslanted at a fixed angle ($45°$) and separate training / test sets were defined for use in experiments.

In the KdK collection, often the digits are connected by ligatures or are broken. Consequently, a strictly connected-component-based approach would fail. We adopted a general segmentation-based approach applicable to complete words or sentences (text lines). The three types of numerical fields used in this paper actually represent an initial testbed for our handwriting recognizer, with augmenting difficulty

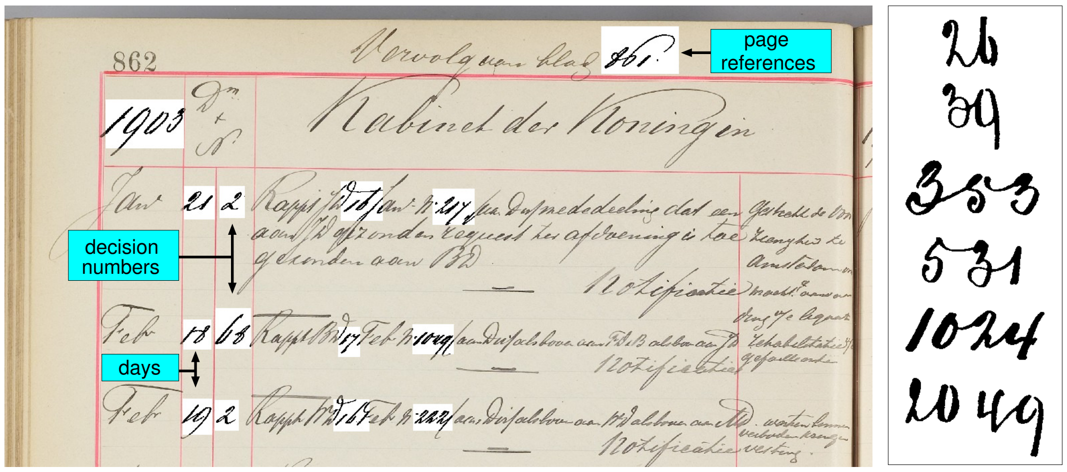

Figure 1. Example of a KdK document showing extensive use of numerical fields (highlighted). We indicate the three types of fields used in the recognition experiments reported in this paper: days, decision numbers and page references. Samples of extracted fields are shown in the right panel to emphasize the need for adopting a generic cursive handwriting recognition approach.

due to the increasing pattern complexity and size of the recognition lexicon: days ('1' to '31'), decision numbers ('1' to '300') and page references ('1' to '3000').

3. Heuristic over-segmentation - Viterbi search

We built our handwriting recognizer following a segmentation-based (split-and-merge) approach [5]. Three processing stages are involved (see Fig. 2):

1) Over-segmentation: a heuristic method is used to define a larger set of potential segmentation points with the goal that it contains, as a subset, the real segmentation points between the characters. The ink is therefore split into graphical entities (graphemes) that are smaller than a complete character. Often the upper / lower word contours are used for this purpose. In our case, the ink is projected onto the horizontal axis and the minima of the smoothed profile define the possible segmentation points. These points are ordered from left to right and are used to define the nodes in a graph. The different combinations of ink segments that can form a letter are encoded in the arcs of this graph (e.g. from node i to node n with $1 \leq n - i \leq 5$). We use min / max widths to define these directed links, which we encode in the adjacency matrix of the segmentation graph.

2) Character recognition: all candidate characters, as defined by the links of the segmentation graph, are passed to a classifier that must generate recognition scores for all character classes involved. The recognition scores $rec[i, n, char]$ can be posterior (log)probabilities or pattern matching distances. We adopt here a formalism based on additive costs that are minimized for the correct class. Any character recognizer can be used at this stage. We will compare two neural networks: a multilayer perceptron (MLP) and a radial basis function network (RBF). Not only the recognition performance, but also the rejection performance of the character classifier is important, because the majority of the pattern candidates obtained by merging ink segments do not correspond to real characters.

As input to the neural networks, we use normalized 28x28 bitmaps (MNIST) [5]. Character deslanting using the principal axis is followed by shifting the center of mass of the ink to the center of the 28x28 template. The same number of units (784-200-10) was used for both networks.

3) Left-to-right search: the best (minimum cost) sequence of characters matching the input image is found using dynamic programming (Viterbi search). Two situations can be distinguished here, depending on whether or not a lexicon is used to constrain the recognition result.

When no lexicon is used, it is sufficient to store only the best recognition result for each character candidate $mrec[i, n] = min_{char}(rec[i, n, char])$ together with its class label. A minimum Viterbi cost $vcost[n]$ is then computed iteratively at each node n of the segmentation graph in left-to-right order (see Fig. 3):

$$vcost[n] = \min_{i \in Inc(n)} (vcost[i] + mrec[i, n]) \qquad (1)$$

$$pback[n] = \underset{i \in Inc(n)}{argmin}(vcost[i] + mrec[i, n]) \qquad (2)$$

where $Inc(n)$ is the set of incoming links to node n. The Viterbi costs must in fact be normalized with respect to the path length in order to avoid the tendency towards shorter solutions that naturally accumulate smaller costs. The optimal path is found by following the backtracking pointers $pback[n]$ from the end node to the begin node of the segmentation graph. The recognition string is obtained by reading off the labels along the Viterbi path. The correct segmentation is thus a byproduct of the recognition process. This lexicon-free approach is relevant for sentence recognition, where an exhaustive lexicon cannot be assumed.

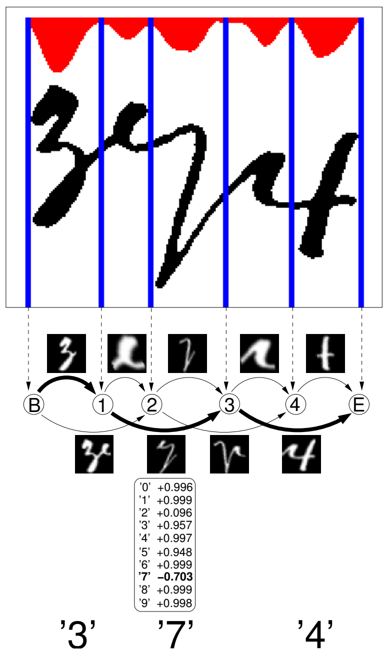

Figure 2. Schematic description of the segmentation-based handwriting recognizer applied to a numerical field.

Often, a lexicon containing all the legal words is available and can be used to constrain the recognition results yielding improved performance. The optimal Viterbi cost must be computed for all the words in the dictionary (eqs 1 and 2 must be modified to take account of character levels and labels). The best matching word with the minimum cost is selected as the final recognition result.

In the lexicon-driven approach, the search for the optimal solution can become the bottleneck, especially for large lexica and long word zones. Attaining speed without compromising performance is possible by organizing the lexicon as a tree (see Fig. 4). In this way, repeated computation is avoided for the common prefix shared by several words.

The segmentation graph and the lexical tree are two discrete structures and the search for the optimal path can be performed using an elegant recursive algorithm (Alg. 1) that enforces an ordered one-to-one correspondence between the tree nodes and a subset of the graph nodes.

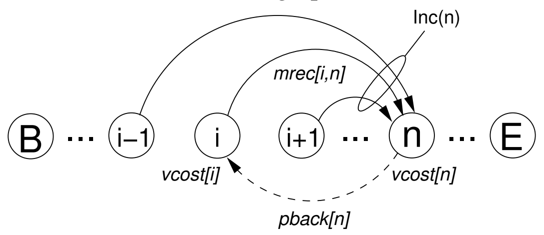

Figure 3. Lexicon-free Viterbi search for the optimal (minimum cost) recognition path (see eqs 1 and 2).

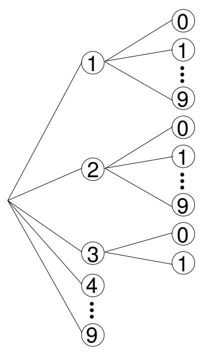

Figure 4. Tree organization of the recognition lexicon: '1' to '31' used for the day (of the month) fields.

In the results section, we will compare the recognition rate obtained by lexicon-free versus lexicon-driven search. We are also interested in the speed up obtained from using a lexical tree versus a flat lexicon.

4. Digit morphing

In contrast with the EM algorithm (Baum-Welch) for HMMs, training the basic character recognizer for a segmentation-based handwriting recognition system is a tricky issue without a standard solution. Our approach was to collect a base set of digit images segmented by hand and to augment this data by generating synthetic examples using random geometric distortions. We were incited by the record performance in digit recognition reported in [7].

Algorithm 1 Depth-first recursive search for the minimal-cost path complying both with the lexical tree and the segmentation graph. In the first call of the recursive function, $tnode$ is the root of the lexical tree, $gnode$ is the first node of the segmentation graph and $cost = 0$. The $finalcost$ array is initialized to ∞.

```
function rsearch($tnode, gnode, cost$)
  /* Recursion stops - cost minimization */
  if isleaf($tnode$) and isend($gnode$) then
    if $cost < finalcost[tnode.word]$ then
      $finalcost[tnode.word] = cost$
    end if
    return
  end if
  /* Move to the next node in the tree and the graph */
  for all $tchild$ in $tnode.children$ do
    for all $gnext$ in $gnode.links$ do
      /* Cost accumulation */
      $newcost = cost + rec[gnode, gnext, tchild.char]$
      /* Search the rest of the word - recursive call */
      rsearch($tchild, gnext, newcost$)
    end for
  end for
end function
```

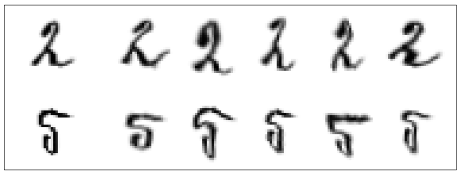

Figure 5. Examples of morphed digits used in training: the original pattern on the left, followed by five randomly distorted versions (28x28 normalized bitmaps).

For every pixel (i, j) of the template image, a random displacement vector $(\Delta x, \Delta y)$ is generated. The displacement field of the complete image is smoothed using a Gaussian convolution kernel with standard deviation σ. The field is finally rescaled to an average amplitude A. The new morphed image $(i^/, j^/)$ is generated using the displacement field and bilinear interpolation $i^/ = i + \Delta x$, $j^/ = j + \Delta y$. This morphing process is controlled by two parameters: the smoothing radius σ and the average pixel displacement A. Both parameters σ and A are measured in units of pixels.

An intuitive interpretation is to imagine that the characters are written on a rubber sheet and we apply non-uniform random local distortions, contracting one part, while maybe expanding another part of the character (see Fig. 5). This random elastic morphing is more general than affine transforms, providing a rich ensemble of shape variations. We applied it to our base set of labeled digits ($\sim$130 samples per class) to obtain a much expanded training dataset (from 1 up to 80 times). The expansion factor f controls the amount of synthetic data: for every base example, $f - 1$ additional morphed patterns are generated and used in training.

This is a cheap method relying on random numbers and basic computer graphics. In this way, a virtually infinite volume of training samples can be fabricated. This stratagem is very successful and does not increase the load at recognition time for parametric classifiers. Essentially, we turn the tables around and, instead of trying to recognize a character garbled in an unpredictable way by the writer in the instantaneous act of handwriting, we generate the deformations ourselves, while training a neural network to become immune to such distortions.

One important question is whether the artificial patterns look natural and are therefore statistically representative for the normal perturbations encountered in handwriting. An empirical answer can be given showing that, for a reasonable choice of morphing parameters, using synthetic training data does increase the final recognition performance.

5. Results

In this section, we analyze how the recognition rate at the level of complete numerical fields is influenced by several factors. The tests were performed on 6740 numerical fields extracted from the KdK book of the year 1903 ($\sim$1000 pages produced by a single writer).

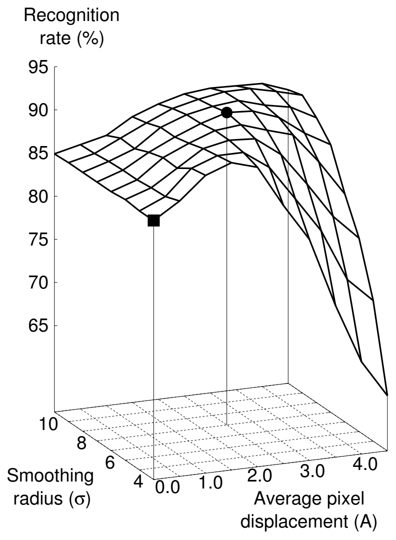

Figure 6. Surface plot showing the recognition rate as a function of the morphing parameters σ and A (for an MLP digit recognizer and $f = 50$): the improvement is from the square mark (no morphing) to the round mark (operational point used further in our experiments).

Extensive retraining-recognition experiments were run in order to determine how the performance depends on the morphing parameters σ and A. Fig. 6 clearly shows that using morphing during training significantly improves performance with respect to the baseline where morphing is not used. The broad peak attests that the choice of the optimal morphing parameters is not critical. We used $\sigma = 8$ and $A = 2.5$ further in our experiments. The corresponding operation point is marked on the surface plot.

As can be seen on the right part of the figure, there is also the potential of destroying performance by using large displacements with reduced smoothing: the patterns are simply pulverized by the strong and uncorrelated distortion field.

In an experiment run on the MNIST digit set, the error rate was reduced from 1.4% to 0.9% by using morphing in training an MLP with the same architecture as the one used on the KdK data (confirming the results reported in [7]).

Fig. 7 shows how the recognition rate depends on the amount of synthetic data used to train the digit recognizer. Improved performance is obtained both for the MLP and RBF, whether or not a lexicon is used in recognition. Initially the RBF performs better than the MLP, but the situation is reversed with increasing amounts of artificial data. The MLP takes better advantage of the examples generated by morphing. All curves from Fig. 7 show that beyond an expansion factor of about 50, a saturation is reached and adding more training data by digit morphing does not result in significant improvements of the numerical field recognition rate. Using a lexicon to constrain the recognition gives an improvement of $\sim$3% compared to lexicon-free results.

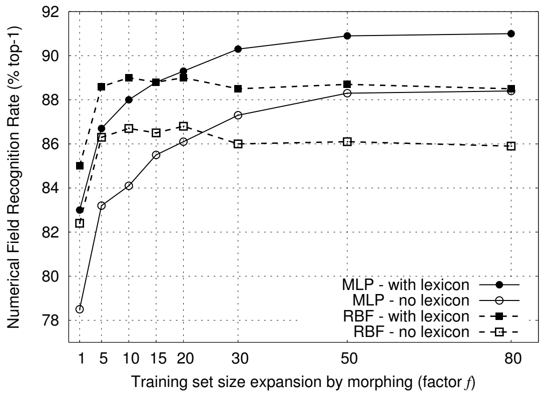

Figure 7. Top-1 recognition rate versus training set expansion factor f. Two digit classifiers (MLP and RBF) are used in two settings (with / without a recognition lexicon).

The recognizer is about 2x faster when recursive search is used with a tree organization of the lexicon as compared to a flat lexicon. The complete test dataset is recognized in approx 20 seconds on a present day CPU (Xeon 1.9 GHz). The speed gains will become important in word recognition with a large lexicon (50 - 100 thousand words).

Table 1 gives the final recognition percentages achieved for the three types of numerical fields extracted from the KdK images (see Fig. 1). The recognition rate decreases with increasing length / lexicon size of the numerical field. The RBF is outperformed by the MLP, which realizes a gain of $\sim$8% due to morphing (top-1 $\sim$91%). In our case, the top-5 recognition rate ($\sim$97%) is an important indicator, as the recognition results will be used for document retrieval in a Google-like search engine for the KdK collection.

Applying the recognizer described here on handwritten words requires investigation of a number of possible improvements: using ascender / descender information, using feature extraction rather than normalized bitmaps for character recognition, improving the reject performance by introducing an additional reject unit in the output layer of the network with *softmax* output normalization, global discriminative training by alternating character segmentation by forced-alignment with neural network training sessions. Results will be reported in upcoming papers.

6. Conclusions

The empirical results have shown that it is beneficial to use synthetic data to train the character recognizer of a segmentation-based handwriting recognition system. We used a smooth random deformation field emulating local elastic distortions of the digit patterns to generate additional data expanding the training set up to 80 times. An MLP benefits more from the morphed samples. By using a lexicon to constrain the results, we reached, on realistic data extracted from a large single-writer historical collection, recognition rates that are usable in document retrieval.

Table 1. Recognition rate on whole numerical fields (extracted from the KdK documents - year 1903). The same search framework is used (employing a lexical tree) in comparing two different digit classifiers (MLP and RBF) trained on a dataset expanded $f = 50$ times using morphed digits.

Num. field (lexicon size)	No. of instances	Recog. rate (%) top-1 / top-5 RBF	MLP
days (31)	2921	92.0 98.9	95.1 99.6
decision nos (300)	2897	91.8 97.9	91.5 98.5
page refs (3000)	922	68.6 84.9	75.7 86.8
Overall	6740	88.7 96.6	90.9 97.3
Improvement due to morphing		+3.7 +2.1	+7.9 +3.2

Efficient recognition of handwritten numerical fields is essential in several concrete applications: reading post codes for mail sorting, reading courtesy amounts for bank check processing, reading numerical identifiers for form recognition. The experience reported in this paper has also bearing to general handwritten word or sentence recognition and the relevance extends to the wider area of historical document analysis as, quite often, these bureaucratic archives make intense use of numerical references and dates.

References

[1] NL-HaNA, 2.02.14, Archief van het Kabinet der Koningin, Den Haag (Netherlands), year 1903.

[2] H. Baird. The state of the art of document image degradation modeling. In *Proc. of 4th DAS*, pp 1–16, Rio de Janeiro, Brazil, 2000.

[3] M. Bulacu, R. van Koert, L. Schomaker, and T. van der Zant. Layout analysis of handwritten historical documents for searching the archive of the Cabinet of the Dutch Queen. In *Proc. of 9th ICDAR*, pp 357–361, Curitiba, Brazil, 2007.

[4] G. Kim and V. Govindaraju. A lexicon driven approach to handwritten word recognition for real-time applications. *IEEE Trans. on PAMI*, 19(4):366–379, 1997.

[5] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998.

[6] U. Pal, K. Roy, and F. Kimura. A lexicon driven method for unconstrained Bangla handwritten word recognition. In *Proc. of 10th IWFHR*, pp 601–606, La Baule, France, 2006.

[7] P. Simard, D. Steinkraus, and J. Platt. Best practices for convolutional neural networks applied to visual document analysis. In *Proc. of 7th ICDAR*, pp 958–962, Edinburgh, Scotland, 2003.

[8] Y. Tay, P. Lallican, M. Khalid, C. Viard-Gaudin, and S. Knerr. An analytical handwritten word recognition system with word-level discriminant training. In *Proc. of 6th ICDAR*, pp 726–730, Seattle, USA, 2001.

[9] T. Varga and H. Bunke. Offline handwriting recognition using synthetic training data produced by means of a geometrical distortion model. *IJPRAI*, 18(7):1285–1302, 2004.