A lexicon-driven approach for optimal segment combination in off-line recognition of unconstrained handwritten Korean words

Soo H. Kim\textsuperscript{a,*}, S. Jeong\textsuperscript{a}, Ching Y. Suen\textsuperscript{b}

\textsuperscript{a}Department of Computer Science, Chonnam National University, 300 Yongbong-dong, Buk-gu, Kwangju 500-757, South Korea
\textsuperscript{b}Center for Pattern Recognition and Machine Intelligence, Concordia University, Suite GM606, 1455 de Maisonneuve Blvd. West, Montreal, Quebec, Canada H3G 1M8

Received 14 May 1999; received in revised form 7 June 2000; accepted 7 June 2000

Abstract

We propose a new method for off-line recognition of unconstrained handwritten words consisting of Korean and numeric characters. To overcome the difficulty in separating touching characters, we adopt an over-segmentation strategy. Given a slice of the input word image, we find the optimal segment combination using a lexicon-driven word scoring technique and a nearest-neighbor classifier. The optimal combination gives the final segmentation positions for individual characters, along with the best matching word in the lexicon. Superiority of the proposed system has been proven by testing it with 908 images of unconstrained words handwritten on live mail pieces.

Keywords: Handwritten Korean words; Over-segmentation; Segment combination; Nearest-neighbor classifier; Lexicon-driven word scoring

1. Introduction

Off-line recognition of unconstrained handwritten words is one of the hottest subjects in pattern recognition today. This technology has a number of applications such as reading addresses on mail pieces [1], reading legal amounts on bank checks [2], processing of tax forms [3], routing FAX messages [4], and so on.

Hundreds of approaches have been reported so far for Roman-style cursive script recognition [5], but only a few trials are published to recognize handwritten Korean words [6]. Our research has been motivated from the fact that most Korean word recognition systems to date suffer from narrow applicability and approaches for English word recognition are not suitable for the recognition of Korean words.

1.1. Related work

Among a number of English word recognition approaches, two have emerged with general popularity: hidden Markov model (HMM)-based approaches [4,7–10], and segmentation-based dynamic programming (DP) techniques [3,11–13]. Both of them take the same segmentation strategy where an input word image is sliced into small segments representing a partial or full character image. They use this over-segmentation strategy to solve the problem of separating touching characters.

The HMM-based approaches are not adequate for recognizing Korean words, since the recognition model can grow very fast and become intractable due to the formation of thousands of states and observation symbols which indicate Korean characters or their partial images. It means that millions of HMM parameters have
to be estimated, and hence results in a need of an impractically large amount of training data. Similarly, the DP technique is not efficient for recognizing Korean words, because it has to examine a lot of segments or groups of consecutive segments, each of which is regarded as a character image to be matched with given character models. In other words, the time complexity of this technique is proportional to the number $C$ of character classes at hand. In case of English words, the value of $C$ is at most 52, but it could be more than 2000 in Korean words.

Most previous studies on Korean word recognition aimed at reading addresses [14–17], since it is possible to construct a highly accurate word recognition system by post-processing with a lexicon. However, so far no systems can recognize totally unconstrained handwritten words because they all assume that the input word image can be separated into individual characters or character components (graphemes). Hence the above methods cannot be used in real applications.

1.2. Proposed approach

In this paper, we propose a new method to recognize unconstrained handwritten words consisting of Korean and numeric characters. To cope with the difficulty in separating touching characters we adopt an over-segmentation strategy, in which a character image is sliced vertically into small segments. Given a slice of the input word image, we find the optimal segment combination using a lexicon-driven word scoring technique and a nearest-neighbor classifier. The optimal combination gives the final segmentation positions for individual characters in the word image, along with the best matching word in the lexicon.

One of the major contributions of our research is to lighten the computational burden that occurs when a conventional segment-then-recognize technique is applied to Korean word recognition. We have devised some speed-up strategies based on the unique characteristics of Korean words and characters, e.g.,

- all Korean characters have almost the same width;
- Korean words are composed of a few characters — two to five in general;
- the number of character classes appearing in a specific character position of words in a lexicon is far smaller than that of all the classes in the lexicon.

The first characteristic can be used for pruning unnecessary segment combinations since a possible character image in a segment combination cannot be much wider than the others. Based on the second characteristic, we can partition a large lexicon into a couple of smaller ones and hence reduce the complexity of lexicon-based operations via a divide-and-conquer strategy. The last characteristic can drastically reduce the complexity of the image-to-prototype matching process. In a conventional segment-then-recognize technique, a possible character image in a segment combination is compared to the models of all character classes in the lexicon. But we use a unigram, defined as a set of character classes appearing in a specific character position of words in a lexicon, to reduce the number of classes to be considered.

In addition, we have tried to use the lexicon as early as possible in our system to avoid unnecessary computation in each processing step. All these speed-up strategies make our system suitable for real-time applications. Superiority of the proposed system has been proven by testing it with 908 images of unconstrained words handwritten on live mail pieces. Two different versions of the system have been evaluated in the experiment: one is fast (0.27 s/image) but has a lower word recognition accuracy of 86.56%, while the other recognizes more accurately at 91.96% but is about 10 times slower (2.83 s/image).

2. Proposed word recognition system

We propose a new lexicon-driven approach for offline recognition of unconstrained handwritten Korean words. The system depicted in Fig. 1 takes a segment-then-recognize approach as the basic paradigm and considers an over-segmentation to overcome the difficulty in separating touching characters in word images. In our over-segmentation technique, a word image is sliced vertically into small segments representing partial or possibly full characters. The system yields an ordered list of words in the lexicon along with the matching scores between the sliced segments and all the lexicon words.

---

Fig. 1. System structure.
There are several lexicons in our system, each of which contains a set of words of the same length. If the minimum and maximum word lengths are $N_{\text{min}}$ and $N_{\text{max}}$, respectively, the number of lexicons becomes $N_{\text{max}} - N_{\text{min}} + 1$ and each is denoted by $L_i$, $N_{\text{min}} \leq i \leq N_{\text{max}}$. The input to our system should be an image of a word in these lexicons to take its class membership as one of word codes in the lexicons. The segmented image is compared repeatedly with every word in the lexicons. In comparing the segmented image $S_1S_2 \ldots S_M$ with a word of length $N$ ($N \leq M$), we find the optimal grouping of $S_1S_2 \ldots S_M$ into $N$ groups, for which the OCR produces the highest matching score. There are many possible cases for combining $M$ segments into $N$ groups, and the combination generation module finds all such segment combinations.

The group filtering module finds out a set of non-redundant groups in all the combinations, because there exist many segment groups whose constituent segments are the same and hence it can lead to fairly overlapped computation of feature extraction and character classification. The feature extraction and character classification modules are performed on these non-redundant groups, and the word scoring module finds the best matching word from the lexicons based on the OCR scores for the segment groups in every combination.

2.1. Vertical slicing

In this module an input word image containing $N$ characters is split vertically into $M$ ($\geq N$) segments, $S_1S_2 \ldots S_M$. We have designed a simple slicing algorithm. It first detects sequentially pre-slicing points by considering the white spaces in the vertical projection of input image and separates the word image into pre-segments at those points. Next, a heuristic rule is applied to all the pre-segments: if the width of a pre-segment is greater than a fixed threshold $SEG_T$, the pre-segment is divided vertically into several smaller segments with the same width as the threshold. Fig. 2 shows an example of vertical slicing, where an image containing three characters is split into nine segments $S_1S_2 \ldots S_9$.

It is important to be able to split a given word image at borders between characters for the next module to group one or more segments into complete character images. So, we impose the following two assumptions on segments $S_m$, $m = 1, \ldots, M$.

1. Any segment $S_m$ represents a partial or full character.
2. The number of segments into which a character may be spliced cannot exceed a fixed threshold $\alpha$.

The first assumption forms the theoretical background of the proposed system, and it indicates that the segmentation should be achievable among all the characters in a word image. If this assumption is satisfied, optimal segment groups corresponding to individual characters can be constructed in the combination generation module, the OCR produces high responses at the segment groups, and then they strongly support the same word code in the lexicon as that of the input image. Even when the assumption is not satisfied, we have observed that our vertical slicing algorithm is still effective in finding the correct lexicon word by using a small value of $SEG_T$.

The second assumption avoids the generation of extra-ordinary segment groups in the following module. Since all Korean characters have almost the same width and hence the width of a character image cannot be much wider than that of any others, the second assumption can be easily met. The value $\alpha$ which limits the number of segments forming a character image plays an important role to improve the efficiency of the proposed system in terms of both speed and recognition accuracy. A smaller value of $\alpha$ is better because the system generates a small number of segment combinations. Here the value $\alpha$ is determined by the value of $SEG_T$ and a somewhat larger value of $SEG_T$ is preferred — to yield a smaller $\alpha$. But it increases the probability of breaking the first assumption. As a compromise, we have chosen the values of $\alpha$ and $SEG_T$ as 5 and 10 pixels, respectively, by trial and error.

2.2. Combination generation

A group or a segment group is a set of one or more consecutive segments, and a combination is a set of consecutive groups. A group will be regarded as the representation of a character image, while a combination with $N$ groups will be matched against the words in a lexicon $L_i$. The combination generation module finds all possible combinations for all possible lengths of words.

The values $M$ and $N$ should be known before performing the combination generation. The value $M$ is provided by the vertical slicing module and the maximum and minimum numbers $N_{\text{max}}$ and $N_{\text{min}}$ are extracted from the lexicons. Since Korean words are composed of quite a few characters — two to five in general, henceforth we assume that $N_{\text{min}}$ is 2 and $N_{\text{max}}$ is 5.

A combination made up of $N$ groups can be expressed as a function that indicates at which $N$ segment groups end. The function can be defined as follows:

![Fig. 2. Example of vertical slicing: (a) input word image, (b) sliced image.](image-url)
$f_N: \{0, 1, 2, \ldots, N\} \rightarrow \{0, 1, 2, \ldots, M\}$,
where
1. $N \leq M$,
2. $f_N(0) = 0$, $f_N(N) = M$, and
3. $f_N(n) - f_N(n - 1) \leq x$, $\forall n \in \{1, 2, \ldots, N\}$,

where 0 in both sets is a null (null group or null segment), and the other elements indicate indices of groups and segments for the input image, respectively. Condition (3) comes from the assumption that a character can be split into at most $x$ segments. The $n$th group $G_n$ in a combination of size $N$ can be represented as

$$G_n = S_{f_N(n-1)+1} \cdots S_{f_N(n)}, \quad 1 \leq n \leq N.$$

When we let a set of all possible combinations consisting of $N$ groups $G_1 \ldots G_N$ be $F_{N|M}$, the goal of this module is to detect all possible combinations $F_{N|M}$, $N_{\min} \leq N \leq N_{\max}$. A dynamic programming method is used as follows to find the combinations in $O(NM)$ time.

**Step 1: Initialization**

$$F_{N|M} = \emptyset, \quad N > m.$$

**Step 2: Recursion for $N_{\min} \leq N \leq N_{\max}$ and $N \leq m \leq M$**

$$F_{N|M} = F_{N|m-1} \otimes m \cup F_{N-1|m-1} \oplus m,$$

where, $F_{N|m} = \{f_N: \{0, 1, \ldots, N\} \rightarrow \{0, 1, \ldots, m\}|f_N(0) = 0, f_N(N) = m, f_N(n) - f_N(n - 1) \leq x\}$.

**Step 3: Output**

$$F_{N|M}, \quad N_{\min} \leq N \leq N_{\max},$$

where $F_{N|m-1} \otimes m$ means modifying every $f_N(N)$ in $F_{N|m-1}$ from $m - 1$ into $m$, and $F_{N-1|m-1} \oplus m$ means equating $f_N(i)$ in $F_{N-1|m-1}$ to $f_N(i)$, $1 \leq i \leq N - 1$, and adding $f_N(N) = m$ at the end of every combination.

The number of combinations is closely related to the speed of the proposed system since it finds the best match by calculating matching scores between every word in the lexicons and every combination generated from the input image. The number can be reduced remarkably by considering $x$, the maximum number of segments that a character can be split. Fig. 3 shows all possible combinations $F_{N|M}$ generated from the segments in Fig. 2(b), where $2 \leq N \leq 5$.

### 2.3. Group filtering

Once all the possible combinations are generated, the words in the lexicon are taken one by one and compared with all the combinations to find the best matched combination with that word. To determine the best combination, the OCR scores are used for constituent groups with respect to the corresponding character classes in the lexicon word. In other words, every group in every combination should be matched against all the relevant character classes in the lexicon words before the combination-to-word matching.

However, there exist many groups whose constituent segments are the same among all the combinations. This may result in possibly redundant feature extraction and character recognition for the same group. The objective of group filtering module is to avoid these redundant calculations by finding out a set of non-redundant groups from all the combinations. Let us define a non-redundant group as $G(k)$, $1 \leq k \leq N_{G(k)}$ where $N_{G(k)}$ is the number of non-redundant groups. We have observed that the number $N_{G(k)}$ is much smaller than the total number of groups in all the combinations, and hence the computational complexity of both the feature extraction and the character recognition modules can be reduced considerably — see Section 3.2 for more details.

We find the set of non-redundant groups by a simple linear search of the combinations. When a non-redundant group $G(k)$ is found, information about the position of the group is maintained. This is because we would like to match $G(k)$ with only the character classes appearing in the same position of the lexicon words. In this regard we define a unigram $l(i)$ as a set of character classes appearing in the $i$th position of all words of length $i$ $(1 \leq i \leq l)$. We store such position information in $\text{MASK}(G(k)) = \{m_{ij}\}$, $N_{\min} \leq i \leq N_{\max}, 1 \leq j \leq N_{\max}$. If $m_{ij} = 1$, $G(k)$ appears in the $j$th position of the combination consisting of $i$ groups. Fig. 4 shows some examples of non-redundant groups associated with corresponding

**Fig. 3.** All possible combinations generated with the segments in Fig. 2(b).

**Fig. 4.** Non-redundant groups among the combinations in Fig. 3, and associated $\text{MASKs}$. 
2.4. Feature extraction

Given a binary image, a MASK column of the images. The feature value from the cell at $p$ vertical, and the four directions considered are horizontal, vertical, left-diagonal, and right-diagonal. Consequently, each non-redundant statistic of aspect ratios of height versus width of Korean character images. The dimension of mesh (values of $P$ and $Q$) affects the recognition performance of character classifier. If the values of $P$ and $Q$ increase, the recognition accuracy increases but the processing speed decreases. In this paper, a $9 \times 7$ mesh is selected based on this trade-off and the statistics of aspect ratios of height versus width of Korean character images. Consequently, each non-redundant group image is transformed into a 252($4 \times 7 \times 7$)-dimensional feature vector.

2.5. Character recognition

Next, we compute class-matching scores for each non-redundant group image. Character classes considered for a group image are determined from the position of combination in which the group image appears. These character classes can be obtained from the MASK associated with the group. In other words, we can get the character classes by a union of the classes in the unigrams that the MASK indicates. The class-matching score for a group image $G(k)$ is calculated as

$$d(C, G(k); C \in \{l_i(j)\} | \text{MASK}(G(k)) = \{m_{ij}\}, \text{ and } m_{ij} = 1\},$$

where $l_i(j)$ represents the unigram consisting of characters appearing in the $i$th position of the words in a lexicon $L_i$, which contains the words made up of $i$ characters. By matching a group image with the classes in the unigrams, instead of all the classes at hand, we can reduce drastically the number of distance calculations — see Section 3.4 for more details.

Two kinds of character classification schemes to implement the above $d(C, G(k))$ function have been adapted in this work: one is MDC (minimum distance classifier) with LVQ (learning vector quantization) prototypes [19], and the other is a modified $k$-NN ($k$-nearest-neighbor) classifier. There is one LVQ prototype per class in the MDC. Given a group image $G(k)$, the MDC simply computes $d(C, G(k))$ as the distance from the feature vector of $G(k)$ and the LVQ prototype for class $C$. On the other hand, the $k$-NN classifier takes all the training samples as prototypes. It finds $k$ nearest prototypes from class $C$ to the feature of $G(k)$ and then computes $d(C, G(k))$ as an average of the $k$ distances. We have observed that the modified $k$-NN classifier shows superior recognition accuracy, while the MDC with LVQ prototypes runs quite fast.

2.6. Word scoring

The eventual goal of our word recognition system is to find the best matching word code for an input image from the lexicons $L_N$, $N_{\min} \leq N \leq N_{\max}$. The system has already split the input word vertically into $M$ segments and then generated all possible combinations via $F_{N|M}$. $N_{\min} \leq N \leq N_{\max}$. Given a word $W$ in a lexicon $L_N$, the following function gives the best matching score of $W$ with respect to all combinations in $F_{N|M}$:

$$D_N(W) = \min_{f \in F_{N|M}} \left\{ \frac{1}{N} \sum_{n=1}^{N} d(C_n, S_{f(n-1)} \ldots S_{f(n)}) \right\},$$

where $W = \{C_1, \ldots, C_N\}$, and $d(\cdot)$ represents the distances from class $C_n$ to $G_n = S_{f(n-1)} \ldots S_{f(n)}$. The best matching word for the input image is determined as the word with the minimum matching score $D$ defined as

$$D = \min_{N_{\min} \leq N \leq N_{\max}} \min_{W \in L_N} D_N(W).$$

Fig. 6 shows an example word scoring. Given a set of segment combinations $F_{N|M}(2 \leq N \leq 5)$ which have been generated with $M$ segments from the vertical slicing module, we calculate every matching score between a combination in $F_{N|M}$ and a word $W$ in $L_N(2 \leq N \leq 5)$. The best matching score for the word $W$ is given by $D_N(W)$ and all
the lexicon words are ranked according to these best matching scores.

3. Experimental results

3.1. Environment

We have implemented the proposed system using a C programming language on a Pentium 350 MHz PC and tested with 908 word images extracted from live mail pieces. The mail pieces were scanned at a resolution of 200 DPI (dots per inch), and the word images were extracted from address regions by a simple word segmentation algorithm and labeled manually. The number of constituent characters within a word image varies from 2 to 5 as summarized in Table 1. Fig. 7 shows some examples of the test images.

The lexicon of our system contains all the strings labeled on the test images. We have constructed the lexicon with four sub-lexicons $L_2$, $L_3$, $L_4$, and $L_5$ according to the length of words they contain, see Fig. 8. Given a lexicon $L_i \ (2 \leq i \leq 5)$, $i$ unigrams denoted as $l_i(1), \ldots, l_i(i)$ are constructed. Table 2 shows the size of four lexicons and 14 associated unigrams. Note that all the lexicon words consist of 362 different character classes: 352 Korean characters and 10 numerals.

The training of our system is very simple because only the prototypes used by the character recognition

Table 1

<table>
<thead>
<tr>
<th>No. constituent characters</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>Total</th>
</tr>
</thead>
<tbody>
<tr>
<td>No. test images</td>
<td>148</td>
<td>507</td>
<td>164</td>
<td>89</td>
<td>908</td>
</tr>
</tbody>
</table>
module are to be learned. We need two kinds of character prototypes for two different character classifiers in our experiment. The MDC is provided with LVQ prototypes, one for each character class. The \(k\)-NN classifier uses all the training samples as prototypes. Prototypes for 352 Korean character classes have been trained with a handwritten Korean character database, namely PE92 [20], and those for 10 numerals are trained with privately collected samples. Table 3 shows the performance of the two classifiers after being trained with the first 70 samples/class and tested with the remaining 30 samples/class.

3.2. Performance of word recognition

Table 4 summarizes the accuracy of our handwritten word recognition system (HWRS in short, hereafter) when tested with 908 word images. The recognition accuracy differs according to the choice of character classifier as well as the number of characters in the image. We have observed that the system always shows better performance when it adapts \(k\)-NN classifier rather than MDC. This is because \(k\)-NN outperforms MDC as shown in Table 3. In addition, the image-to-prototype distances generated by \(k\)-NN are far more reliable than those of

**Table 2**

Size of lexicons and associated unigrams

<table>
<thead>
<tr>
<th></th>
<th>(L_2)</th>
<th>(L_3)</th>
<th>(L_4)</th>
<th>(L_5)</th>
<th>Total</th>
</tr>
</thead>
<tbody>
<tr>
<td>No. of words</td>
<td>39</td>
<td>180</td>
<td>50</td>
<td>44</td>
<td>313</td>
</tr>
<tr>
<td>No. of characters</td>
<td>L(1) 28</td>
<td>L(2) 21</td>
<td>L(3) 11</td>
<td>L(4) 9</td>
<td>L(5) 4</td>
</tr>
</tbody>
</table>

**Table 3**

Performance of MDC and \(k\)-NN classifiers on 362 character classes

<table>
<thead>
<tr>
<th></th>
<th>Cumulative accuracy (%)</th>
<th>Speed (chr/s)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>1st</td>
<td>2nd</td>
</tr>
<tr>
<td>MDC</td>
<td>70.17</td>
<td>78.87</td>
</tr>
<tr>
<td>(k)-NN</td>
<td>73.97</td>
<td>85.82</td>
</tr>
</tbody>
</table>

**Table 4**

Word recognition accuracy for 908 test images

<table>
<thead>
<tr>
<th>No. constituent char's</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>Total</th>
</tr>
</thead>
<tbody>
<tr>
<td>No. input images</td>
<td>148</td>
<td>507</td>
<td>164</td>
<td>89</td>
<td>908</td>
</tr>
<tr>
<td>HWRS with MDC</td>
<td>84.46</td>
<td>86.39</td>
<td>89.02</td>
<td>86.52</td>
<td>86.56</td>
</tr>
<tr>
<td>HWRS with (k)-NN</td>
<td>92.57</td>
<td>91.91</td>
<td>94.74</td>
<td>89.89</td>
<td>91.96</td>
</tr>
</tbody>
</table>
To see the effect of lexicon size on the proposed HWRS, we have observed the performance as the lexicon size varies from 10 to 313. Subset words were chosen randomly from the whole lexicon, and the performance was measured with a subset of 908 word images corresponding to the selected words. This selection-and-test procedure has been performed 10 times for every lexicon size to get unbiased performance by averaging the 10 different measurements. Table 5 summarizes the recognition accuracy on various lexicon sizes, and Fig. 10 shows that the lexicon size affects the performance of HWRS — both the accuracy of top choice and the cumulative accuracy up to the fifth candidates decrease as the lexicon size increases.

Our system can recognize handwritten words consisting of both Korean and numeric characters. So, further analysis on the overall recognition accuracy of 91.96% has been conducted by considering the effect of numeral characters within the word image. Generally, the widths of numeral characters differ from Korean characters, and they are written with larger slants. These facts may cause erroneous segmentation in the vertical slicing module, and hence result in lower recognition accuracy. In Table 6, recognition accuracy for words consisting of only Korean characters (words/KC) and words including at least one numeral character (words/NC) are compared. The same lexicon with 313 words is used in this experiment. As we have predicted, recognition accuracy for words consisting of only Korean characters is a little higher.

Table 7 shows some statistics on the computational complexity of the system. The first column of the table shows the number $M$ of segments generated by the vertical slicing module. In our system, there is no segment whose width exceeds 10 pixels, i.e., we have set the value of $SEG_T$ to 10 in implementing the vertical slicing module. The value $M$ varies from 4 to 24 depending on the manner of touching between characters as well as the image width. The average value of $M$ overall the input images is 10.95.

The fourth column shows the number of combinations generated by the system. The number of combinations for grouping $M$ consecutive segments into $N$ consecutive groups is $\binom{M-1}{N-1}$ (refer to the third column), but we can reduce it by considering $z$ in the combination generation module. The average value of this reduction ratio is 0.4. The fifth column shows the number of non-redundant groups among all the combinations. Only these groups are taken into considerations by the character recognizer. One can see that only 44.27 groups on the average remain after the execution of the group filtering module.

The last column shows the average number of character classes associated with each non-redundant group. A 252-dimensional feature vector computed from each non-redundant group would be compared to the prototypes for this number of classes — its average value is 61. By matching a group with these classes, instead of all the 362 classes at hand, we can reduce the number of distance comparisons of the character recognizer by a factor of 6 (362/61).

### Table 5

<table>
<thead>
<tr>
<th>Lexicon size</th>
<th>10</th>
<th>20</th>
<th>40</th>
<th>80</th>
<th>160</th>
<th>313</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>99.11</td>
<td>97.50</td>
<td>96.67</td>
<td>96.64</td>
<td>94.84</td>
<td>91.96</td>
</tr>
<tr>
<td>Top-N choice</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>99.11</td>
<td>98.65</td>
<td>98.30</td>
<td>98.14</td>
<td>97.81</td>
<td>96.70</td>
</tr>
<tr>
<td>2</td>
<td>99.39</td>
<td>99.09</td>
<td>98.73</td>
<td>98.37</td>
<td>98.18</td>
<td>97.36</td>
</tr>
<tr>
<td>3</td>
<td>99.39</td>
<td>99.09</td>
<td>98.82</td>
<td>98.42</td>
<td>98.41</td>
<td>97.80</td>
</tr>
<tr>
<td>4</td>
<td>99.39</td>
<td>99.09</td>
<td>98.82</td>
<td>98.46</td>
<td>98.43</td>
<td>98.13</td>
</tr>
<tr>
<td>5</td>
<td>99.39</td>
<td>99.09</td>
<td>98.82</td>
<td>98.46</td>
<td>98.43</td>
<td>98.13</td>
</tr>
</tbody>
</table>

### Table 6

<table>
<thead>
<tr>
<th>Top-N-choice</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>No. of test images</th>
</tr>
</thead>
<tbody>
<tr>
<td>Words/KC</td>
<td>92.55</td>
<td>97.05</td>
<td>97.43</td>
<td>97.82</td>
<td>98.07</td>
<td>779</td>
</tr>
<tr>
<td>Words/NC</td>
<td>88.37</td>
<td>94.57</td>
<td>96.90</td>
<td>97.67</td>
<td>98.45</td>
<td>129</td>
</tr>
</tbody>
</table>
Table 7
Statistics on computational complexity

<table>
<thead>
<tr>
<th>Number of segments (M)</th>
<th>Number of test images</th>
<th>Number of combinations (without z)</th>
<th>Number of combinations (with z)</th>
<th>Number of non-redundant groups</th>
<th>Average unigram size</th>
</tr>
</thead>
<tbody>
<tr>
<td>4</td>
<td>1</td>
<td>7</td>
<td>7</td>
<td>10</td>
<td>58.4</td>
</tr>
<tr>
<td>5</td>
<td>16</td>
<td>15</td>
<td>15</td>
<td>15</td>
<td>68.1</td>
</tr>
<tr>
<td>6</td>
<td>51</td>
<td>30</td>
<td>30</td>
<td>20</td>
<td>78.2</td>
</tr>
<tr>
<td>7</td>
<td>50</td>
<td>56</td>
<td>54</td>
<td>25</td>
<td>85.6</td>
</tr>
<tr>
<td>8</td>
<td>68</td>
<td>98</td>
<td>91</td>
<td>30</td>
<td>84.8</td>
</tr>
<tr>
<td>9</td>
<td>109</td>
<td>162</td>
<td>143</td>
<td>35</td>
<td>80.0</td>
</tr>
<tr>
<td>10</td>
<td>117</td>
<td>255</td>
<td>208</td>
<td>40</td>
<td>72.8</td>
</tr>
<tr>
<td>11</td>
<td>135</td>
<td>385</td>
<td>280</td>
<td>45</td>
<td>64.2</td>
</tr>
<tr>
<td>12</td>
<td>108</td>
<td>561</td>
<td>350</td>
<td>50</td>
<td>53.4</td>
</tr>
<tr>
<td>13</td>
<td>77</td>
<td>793</td>
<td>406</td>
<td>55</td>
<td>44.5</td>
</tr>
<tr>
<td>14</td>
<td>64</td>
<td>1092</td>
<td>436</td>
<td>60</td>
<td>36.9</td>
</tr>
<tr>
<td>15</td>
<td>44</td>
<td>1470</td>
<td>433</td>
<td>65</td>
<td>30.4</td>
</tr>
<tr>
<td>16</td>
<td>32</td>
<td>1940</td>
<td>400</td>
<td>70</td>
<td>24.8</td>
</tr>
<tr>
<td>17</td>
<td>19</td>
<td>2516</td>
<td>340</td>
<td>75</td>
<td>20.5</td>
</tr>
<tr>
<td>18</td>
<td>4</td>
<td>3213</td>
<td>265</td>
<td>76</td>
<td>17.3</td>
</tr>
<tr>
<td>19</td>
<td>6</td>
<td>4047</td>
<td>189</td>
<td>73</td>
<td>15.0</td>
</tr>
<tr>
<td>20</td>
<td>4</td>
<td>5035</td>
<td>122</td>
<td>66</td>
<td>13.2</td>
</tr>
<tr>
<td>23</td>
<td>2</td>
<td>9108</td>
<td>15</td>
<td>24</td>
<td>11.5</td>
</tr>
<tr>
<td>24</td>
<td>1</td>
<td>10902</td>
<td>5</td>
<td>13</td>
<td>11.2</td>
</tr>
</tbody>
</table>

Table 8
Processing time of the system for a word image (unit: ms)

<table>
<thead>
<tr>
<th></th>
<th>HWRS with MDC</th>
<th>HWRS with k-NN</th>
</tr>
</thead>
<tbody>
<tr>
<td>Vertical slicing</td>
<td>0.3</td>
<td>0.3</td>
</tr>
<tr>
<td>Combination generation</td>
<td>72.7</td>
<td>72.7</td>
</tr>
<tr>
<td>Feature extraction</td>
<td>135.3</td>
<td>135.3</td>
</tr>
<tr>
<td>Character recognition</td>
<td>29.0</td>
<td>2594.1</td>
</tr>
<tr>
<td>Word scoring</td>
<td>8.5</td>
<td>8.5</td>
</tr>
<tr>
<td>Misc.</td>
<td>22.2</td>
<td>18.1</td>
</tr>
<tr>
<td>Total</td>
<td>268.0</td>
<td>2829.0</td>
</tr>
</tbody>
</table>

In summary, an input word image is sliced into 10.95 segments, and they are put into 249.37 combinations to obtain 44.27 non-redundant groups, and each group is compared to 61 character classes. In other words, 44.27 times of feature extractions and the same number of character classifications with respect to 61 classes are performed to recognize the input image. Table 8 shows the average processing time for a word image.

3.3. Failure analysis

With our system, 73 out of 908 test images are misclassified. Reasons for the recognition failure can be categorized roughly into two types: one is the error of character recognition and the other results from the segmentation error. We have observed that 44 out of the 73 mis-classifications belong to the first type and the remaining 29 images belong to the second type. A character recognition error occurs when the quality of input image is quite poor, i.e., when the input image contains broken strokes, heavily cursive characters, unusual writing styles, etc. Fig. 11 shows some examples of images of poor quality. Even though our system fails to recognize poor image, however, the correct recognition result appears within the set of a few recognition candidates in most cases — see the cumulative recognition accuracy in Fig. 9.

A segmentation error happens when either of the two assumptions (see Section 2.1) imposed on the segmented images is not satisfied during the vertical slicing — refer to Fig. 12. Among the 29 images which were misclassified due to incorrect segmentation, segmentation results of nine word images do not satisfy the first assumption that a segment is a partial or full character image, and the segments of the remaining 20 images do not satisfy the second assumption. The first assumption is not satisfied when some strokes in two adjacent character images touched at many places or overlapped vertically, and the second assumption is not satisfied when a character...
Fig. 12. Example of incorrect classifications due to segmentation error.

in the image is written much wider than the other characters.

4. Conclusion

We have proposed a system for off-line recognition of handwritten Korean words containing numeric characters. The recognition accuracy of the system for word images of live mail ranges from 86.56 to 91.96% and the processing time for recognizing an image varies from 0.27 to 2.83 s, depending on the choice of character classifier. One of the main ideas in implementing the system is to utilize the distinctive characteristics of Korean words and characters: slicing the input words into small segments to cope with touching characters, generating segment combinations efficiently by a dynamic programming technique, pruning the combinations based on a constraint on character slicing, splitting a large lexicon into smaller ones according to word lengths, and using unigrams for character classifications to reduce the number of character classes to be compared. Although we have presented a performance for word images in mail pieces, our system can recognize any kinds of unconstrained Korean words if a lexicon is provided.

We plan to continue to enhance the performance of this recognition system. With respect to the processing time, it is quite important to reduce the number of segments generated by the vertical slicing module. Since the recognition accuracy of the system is highly dependent on the character classifier, the character classification module should be improved using more representative prototypes. Our work can be applied effectively to build an address interpretation system, composing of a word segmentation algorithm which extracts individual words from handwritten text lines and an efficient search strategy which uses the results of word recognition.

5. Summary

In this paper, we propose a new method to recognize unconstrained handwritten words consisting of Korean and numeric characters. To overcome the difficulty in separating touching characters, we adopt an over-segmentation strategy in which a character image is sliced vertically into small segments. Given a slice of the input word image, we find the optimal segment combination using a lexicon-driven word scoring technique and a nearest-neighbor classifier. The optimal combination gives the final segmentation positions for individual characters in the word image, along with the best matching word in the lexicon.

One of the major contributions of our research is to lighten the computational burden that occurs when a conventional segment-then-recognize technique is applied to recognize Korean words. We have devised some speed-up strategies based on the characteristics of Korean words and characters. First, the characteristic that all Korean characters have almost the same width can be used for pruning unnecessary segment combinations. Second, based on the characteristic that Korean words are composed of a few characters — two to five in general, we partition a large lexicon into a couple of smaller ones and hence reduce the complexity of lexicon-based operations via a divide-and-conquer strategy. Finally, we drastically reduce the complexity of the image-to-prototype matching process using the characteristic that the number of character classes appearing in a specific character position of words in a lexicon is much smaller than that of all the classes in the lexicon. Generally, a possible character image in a segment combination is compared to the models of all character classes in the lexicon. But we use a unigram, defined as a set of character classes appearing in a specific character position of words in a lexicon, to reduce the number of classes to be considered.

In addition, we have tried to use the lexicon as early as possible in our system to avoid unnecessary computation in each processing step. All these speed-up strategies make our system adequate for real-time applications. Superiority of the proposed system has been proven by testing it with 908 images of unconstrained words handwritten on live mail pieces. Two different versions of the system have been evaluated in the experiment: one is fast but has a lower word recognition accuracy of 86.56%, while the other recognizes more accurately at 91.96% but is about 10 times slower.

Acknowledgements

This work was supported by the Brain Korea 21 Project and grant number 98-0102-02-01-3 from the interdisciplinary Research program of the KOSEF.

References

[1] S.N. Srihari, E.J. Keubert, Integration of hand-written address interpretation technology into the United States


About the Author—SOO HYUNG KIM received his B.S. degree in Computer Engineering from Seoul National University in 1986, and his M.S. and Ph.D degrees in Computer Science from Korea Advanced Institute of Science and Technology in 1988 and 1993, respectively. From 1990 to 1996, he was a senior member of research staff in Multimedia Research Center of Samsung Electronics Co., Korea. Since 1997, he has been an assistant professor in the Department of Computer Science, Chonnam National University, Korea. His research interests are handwritten character recognition, computer vision and neural network modeling.

About the Author—S. JEONG received her B.S. and M.S. degree in Statistics from Chonnam National University in 1996 and 1998, respectively. At present, she is a Ph.D. student in the Department of Computer Science of the same university. Her research interest is handwritten character recognition.

About the Author—CHING Y. SUEN joined the Department of Computer Science of Concordia University, Montreal, in 1972. Presently he is the Director of CENPARMI, the Center for Pattern Recognition and Machine Intelligence of Concordia. Prof. Suen is the author/editor of 11 books on subjects ranging from Computer Vision and Shape Recognition, Handwriting Recognition and Expert Systems, to Computational Analysis of Mandarin and Chinese. Dr. Suen is the author of more than 250 papers on these subjects. A fellow of the IEEE, IAPR and the Royal Society of Canada, Dr. Suen is an Associate Editor of several journals related to Pattern Recognition. He is the Past President of the Canadian Image Processing and Pattern Recognition Society, and the Chinese Language Computer Society. He is Founder or Co-founder of several conferences, including Vision Interface, IWFHR, and ICDAR. Prof. Suen is the recipient of several awards for outstanding contributions to pattern recognition, expert systems, and computational linguistics.