Top-Down Likelihood Word Image Generation Model for Holistic Word Recognition

Eiki Ishidera[1], Simon M. Lucas[2], and Andrew C. Downton[3]

[1] Multimedia Research Labs. NEC Corporation
4-1-1, Miyazaki, Miyamae-ku, Kawasaki, 216-8555, Japan
ishide@ccm.cl.nec.co.jp
[2] Dept. of Computer Science
University of Essex, Colchester CO4 3SQ, UK
sml@essex.ac.uk
[3] Dept. of Electronic Systems Engineering
University of Essex, Colchester CO4 3SQ, UK
acd@essex.ac.uk

Abstract. This paper describes a new top-down word image generation model for word recognition. This model can generate a word image with a likelihood based on linguistic knowledge, segmentation and character image. In the recognition process, first, the model generates the word image which approximates an input image best for each of a dictionary of possible words. Next, the model calculates the distance value between the input image and each generated word image. Thus, the proposed method is a type of holistic word recognition method. The effectiveness of the proposed method was evaluated in an experiment using type-written museum archive card images. The difference between a non-holistic method and the proposed method is shown by the evaluation. The small errors accumulate in non-holistic methods during the process carried out, because the non-holistic methods can't cover the whole word image but only part images extracted by segmentation, and the non-holistic method can't eliminate the black pixels intruding in the recognition window from neighboring characters. In the proposed method, we can expect that no such errors will accumulate. Results show that a recognition rate of 99.8% was obtained, compared with only 89.4% for a recently published comparator algorithm.

1 Introduction

Converting old documents, such as museum archive cards, into electronic data is one of the most important applications of OCR techniques [1]. Such archives are typically recorded using poor quality typewriting that dates from the early 20th Century. Depending on such factors as the age of the archive card, and the state of typewriter and its ribbon, archives contain faded or broken characters, or over-heavy touching characters. These make segmentation difficult.

Several methods have been proposed to avoid the segmentation difficulties in poor quality machine-printed documents. Sawaki [2] proposed a method of segmenting and recognizing characters in a newspaper by using displacement matching and a complementary similarity measure. Lucas [3] proposed a method with a sliding OCR window and an efficient graph search algorithm. In the case of handwriting, the method employed by Ozdil [4] can extract each individual character from cursive script. Since these methods do not involve any prior segmentation, they are robust to problems such as broken and touching characters. We classify these methods as the "comprehensive OCR approach". They assume that the region in which the best recognition score is obtained constitutes the best segmentation. However, it is difficult to know how to treat a good recognition score from an area which does not represent the correct single character(s), thus we should not overlook the potential correctness of the segmentation.

There are also other segmentation-free approaches such as the holistic approach[5][6][7] and HMM approaches [8][9][10]. Holistic word matching treats the word as a single, indivisible entity and attempts to recognize it using features of the word as a whole[5][6]. The method employed by Lu [7] generates the word template from individual character templates and then compares the word template to the image to be recognized. This method is not very robust to the case of including partly faded or broken characters because it eliminates all of the space between the adjacent characters in both the template image and the word image. Such an elimination seriously deforms the shape of the image with included faded or broken characters.

In the HMM method, the potential correctness of the segmentation is not used explicitly[8], so the HMM method involves the same problem as the comprehensive OCR approach. Lethelier[9] and Gilloux[10] proposed a probabilistic model based on HMMs by which the potential correctness of the segmentation is represented. However, the method proposed by Lethelier[9] generates some segmentation hypotheses, so this method is not a segmentation-free approach. In the method employed by Gilloux[10], the potential correctness of the segmentation is estimated through the Baum-Welch re-estimation algorithm. All these approaches are however based on bottom-up models.

In this paper we describe a new top-down likelihood word image generation model for word recognition. Since the model proposed in this paper contains three kinds of explicit likelihoods (linguistic knowledge, segmentation and character image), it can generate each word image with a likelihood value of itself. In the recognition process, first, the model generates the word image which approximates an input best over all search parameters. Next, the model calculates the distance value between the input image and the generated word image. The effectiveness of the proposed method was evaluated in an experiment using type-written museum archive card images. Results show that a recognition rate of 99.8% was obtained.

2 Word Image Generation Model

In this section, we define the word image generation model. Let us consider the case of a human writing down a given word on a piece of paper by hand.

First, the word to be written down is generated from a word generator. Next, the style of writing (cursive script or block style) is chosen[11][12] and the position of the writer's hand and the paper roughly gives a location of each individual character. Then the writer can draw each character's pattern on the paper. We can also consider the case that the person types a given word on paper with a typewriter. In this case, the status of the typewriter gives the style (font type) and the location of each character. We can consider that each individual character's pattern appears individually and independently.

The word image generation model proposed here consists of four parts. The first is a word generator, the second a style generator, the third a character locator and the fourth a character image generator (fig 1).

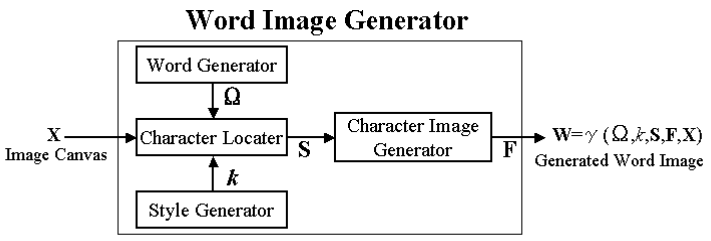


Fig. 1. Word Image Generator

The word generator generates the word $\boldsymbol{\Omega} = \{\omega_1, \omega_2, \cdots, \omega_n\}$ to be written.The word $\boldsymbol{\Omega}$ consists of n characters, and ω_i indicates the i-th character's category. Since we are considering the case that the word $\boldsymbol{\Omega}$ is given, we can assume that $\boldsymbol{\Omega}$ is independent of any other information and thus, the probability of $\boldsymbol{\Omega}$ is set as a constant.

$$P(\boldsymbol{\Omega}) = Const.$$

The style of writing is also specified. In the case of typewriting, for example, the style gives the type of typewriter (font type, the size of the font and the pitch of individual characters), status of the ink ribbon, the strength of typing (heavy or light). We assume that the style is independent of any other information, so the probability of the style "k" can be set as constant.

$$P(k) = Const.$$

Next, let $\boldsymbol{X}$ be an image canvas consisting of width and height. When the word $\boldsymbol{\Omega}$, the style k and the image canvas $\boldsymbol{X}$ is obtained, the character locater gives the location and the size of each character as $\boldsymbol{S} = \{s_1, s_2, \cdots, s_n\}$. Here, $s_i = \{w_i, h_i, x_i, y_i\}$ is the i-th segment which is represented by the size and coordinate parameters. A segment s_i gives the position and size of the character image corresponding to the i-th character's category ω_i. Here, w_i and h_i are the width and the height, and (x_i, y_i) is the center of the segment s_i respectively.

We assume that the size and the location of the character depends only on the category of the character, and the category, size and location of the previous character. Thus, we assume a bi-gram for the character locator, that the i-th segment s_i is correlated only to the (i-1)-th segment s_{i-1}, the i-th character's category ω_i and the (i-1)-th character's category ω_{i-1}. We also assume that the size of the image canvas $\boldsymbol{X}$ roughly gives the size of the word image to be generated. Then, we can write the probability of $\boldsymbol{S}$ as follows.

$$\begin{aligned} P(\boldsymbol{S} \mid \boldsymbol{\Omega}, k, \boldsymbol{X}) &= P(s_1 \mid \boldsymbol{\Omega}, k, \boldsymbol{X}) P(s_2 \mid s_1, \boldsymbol{\Omega}, k, \boldsymbol{X}) \cdots P(s_n \mid s_{n-1}, \boldsymbol{\Omega}, k, \boldsymbol{X}) \\ &= P(s_1 \mid \omega_1, k, \boldsymbol{X}) \prod_{i=2}^{n} P(s_i \mid s_{i-1}, \omega_{i-1}, \omega_i, k, \boldsymbol{X}) \end{aligned}$$

In the last step, the character image generator fills each segment of the postulated word by corresponding character images (normalized in size) according to the style k. When we consider the case of typewriting, we can assume that the character's image is generated independently and individually only according to the law of a probability density function. Thus, we can write the probability of $\boldsymbol{F} = \{f_1, f_2, \cdots, f_n\}$ as follows.

$$P(\boldsymbol{F} \mid \boldsymbol{\Omega}, k, \boldsymbol{S}, \boldsymbol{X}) = \prod_{i=1}^{n} P(f_i \mid \omega_i, k)$$

Here, f_i is the i-th character's image, corresponding to ω_i and s_i.

Then we can obtain the probability of a generated word image $\boldsymbol{W}$ as follows;

$$\begin{aligned} P(\boldsymbol{W} \mid \boldsymbol{X}) &= P(\boldsymbol{\Omega}, k, \boldsymbol{S}, \boldsymbol{F} \mid \boldsymbol{X}) \\ &= P(\boldsymbol{\Omega}) P(k) P(s_1 \mid \omega_1, k, \boldsymbol{X}) P(f_1 \mid \omega_1, k) \\ &\quad \prod_{i=2}^{n} P(f_i \mid \omega_i, k) P(s_i \mid s_{i-1}, \omega_{i-1}, \omega_i, k, \boldsymbol{X}) \end{aligned} \tag{1}$$

Figure 2 shows an example of the case where the word $\boldsymbol{\Omega}$={A,l,b,a,n,y} and the image canvas $\boldsymbol{X}$ are given. In this figure, the first character's category ω_1="A", the first segment s_1 and the first character's image f_1 are shown.

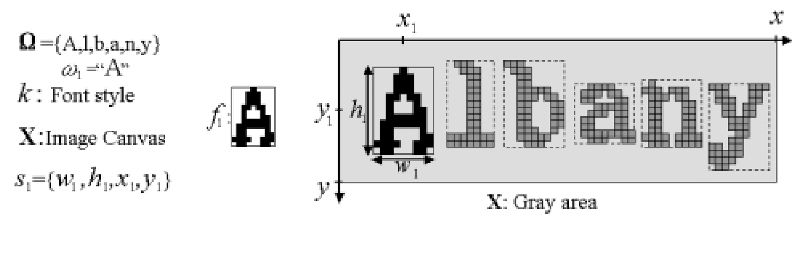


Fig. 2. Example $\boldsymbol{X}$, $\boldsymbol{\Omega}$,$\boldsymbol{S}$ and $\boldsymbol{F}$

3 Word Recognition Algorithm

In this section, we discuss how to recognize the input image $\boldsymbol{Y}$ with the proposed word image generation model. A recognition process can be carried out by searching for the generated word image $\hat{\boldsymbol{W}}$ as shown below.

$$P(\hat{\boldsymbol{W}} \mid \boldsymbol{Y}) = \max_{\mathbf{W}} \frac{P(\boldsymbol{Y} \mid \boldsymbol{W})P(\boldsymbol{W})}{P(\boldsymbol{Y})}$$

To maximize the equation above, we can negrect the term of $P(\boldsymbol{Y})$, we obtain the equation below.

$$\max_{\mathbf{W}} P(\boldsymbol{Y} \mid \boldsymbol{W})P(\boldsymbol{W}) = \max_{\boldsymbol{\Omega},k,\mathbf{S},\mathbf{F}} P(\boldsymbol{Y} \mid \gamma(\boldsymbol{\Omega}, k, \boldsymbol{S}, \boldsymbol{F}, \boldsymbol{X}))P(\gamma(\boldsymbol{\Omega}, k, \boldsymbol{S}, \boldsymbol{F}, \boldsymbol{X})) \tag{2}$$

Here, we set $\boldsymbol{W} = \gamma(\boldsymbol{\Omega}, k, \boldsymbol{S}, \boldsymbol{F}, \boldsymbol{X})$. Since the generated word image $\hat{\boldsymbol{W}}$ is generated from the parameter values of $\boldsymbol{\Omega}$, k, $\boldsymbol{S}$ and $\boldsymbol{F}$, the recognition process will be carried out by searching for parameter values that give the maximum value of the equation (2). To do this, we can generate all possible word images, trying all possible parameter values, and then compare each generated word image to the input image $\boldsymbol{Y}$. However, this is computationally very expensive.

Generating the word images that are obviously different from the input image makes no sense, because such images generated may be useless most of the time and they may not give a maximum value of the equation (2). In this paper, we at first search parameter values which maximize the equation below instead of equation (2), and then generate word image with estimated parameter values.

$$\begin{aligned} P(\boldsymbol{W} \mid \boldsymbol{Y}) = {} & P(\boldsymbol{\Omega})P(k)P(f_1 \mid \omega_1, k)P(s_1 \mid \omega_1, k, f_1, \boldsymbol{Y}) \\ & \prod_{i=2}^{n} P(f_i \mid \omega_i, k)P(s_i \mid s_{i-1}, \omega_{i-1}, \omega_i, k, f_i, \boldsymbol{Y}) \end{aligned} \tag{3}$$

The word recognition algorithm consists of four steps. Since we are considering the case of typewriting, the size of the segment (w_i, h_i) is set as constant.

In the first step, we roughly estimate the position and the size of the first character's segment s_1. In the second step, we estimate the maximum value of the term $P(s_1 \mid \omega_1, k, f_1, \boldsymbol{Y})$ in equation (3). In the third step, we estimate the maximum value of the term $P(s_i \mid s_{i-1}, \omega_{i-1}, \omega_i, k, f_i, \boldsymbol{Y})$ where $(i = 2, \cdots, n)$. Finally, in the fourth step, we generate word image with parameter values estimated above and then calculate the distance value between the input image $\boldsymbol{Y}$ and the generated word image $\boldsymbol{W}$. However, the parameter values estimated in Step1, Step2 and Step3 may lead the local maximim value of equation (2).

Since the states of Ω and k are independent of any other information, this procedure is carried out for all possible words in the dictionary and all defined font styles. Then we can choose the best candidate as the recognition result.

3.1 Step 1

In step 1, we find the left side edge (x_{min}) of the input image $\boldsymbol{Y}$, and the upper edge (y_{min}) within the fixed area as shown in figure 3. Here, the fixed area means a standard character area that approximates the standard width of character (w_s).

We can roughly estimate the position of the first character from the position (x_{min}, y_{min}) with standard width and height of the character. This step allow us to roughly estimate the location s_1.

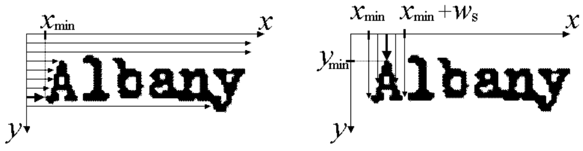


Fig. 3. Finding initial location

3.2 Step 2

In step 2, we find the best location of the first character ω_1. After the initial location s_1 is obtained in step 1 (Figure 3), we optimise this location by moving the template of ω_1 within a fixed area to find the location which gives the best matching (city block distance) between the template and the part image of $\boldsymbol{Y}$ extracted by s_1.

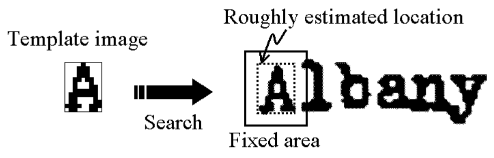


Fig. 4. Example searching area

Figure 4 shows the case where we try to recognize the input "Albany" image as the word "Albany". In figure 4, the dashed box is the location roughly estimated in Step1 and the larger box surrounding it shows the fixed area within which the template is moved. To find the best location means that we estimate s_1 which gives the maximum value $P(s_1 \mid \omega_1, k, f_1, \boldsymbol{Y})$in equation (3).

The size of the s_i (w_i and h_i) is fixed for each joint event of ω_i and k. The area to be searched is set as ± 3 pixels horizontally and ± 4 pixels vertically from the initial position, but these values are variable parameters. In the section on evaluation, we will evaluate various parameter values.

3.3 Step 3

In step 3, first, we roughly assume the location of the second character (s_2) from the location of the first character (s_1) calculated in Step2. The character's category ω_1 and ω_2 is also used for estimating s_2. We can roughly classify the font metrics of characters into two classes. One is the "ascender class" (AC) and the other "non-ascender class"(NAC). The capital characters are classified as ascender class. The lower case characters of "b,d,f,h,i,j,k,l,t" are also classified as ascender class. The non-ascender class includes "a,c,e,g,m,n,o,p,q,r,s,u,v,w,x,y,z".

We can roughly predict the location of the second character's leftside-upper corner (x'_i, y'_i) with the location of the first character's leftside-upper corner and the category of the first character ω_1 and the second character ω_2. The relationship between (x'_{i-1}, y'_{i-1}) and (x'_i, y'_i) is as follows;

$$x'_i = x'_{i-1} + \Delta x$$
$$y'_i = \begin{cases} y'_{i-1} + \Delta y \text{ If } \omega_{i-1} \text{ is "AC" and } \omega_i \text{ "NAC".} \\ y'_{i-1} - \Delta y \text{ If } \omega_{i-1} \text{ is "NAC" and } \omega_i \text{ "AC".} \\ y'_{i-1} \qquad \text{else.} \end{cases}$$

Here, we set the value of Δx at 12, and the value of Δy at 5.

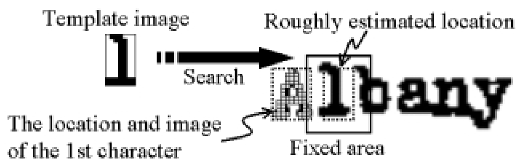


Fig. 5. Example searching area

The location roughly estimated is shown in figure 5 as a dashed box, and gives a rough estimate s_2.

Then, as carried out in step 2, we move the template of ω_2 within a fixed area and find the location which gives the best matching between the template and the part image of $\boldsymbol{Y}$ extracted by s_2. To find the best location of s_2 means that we estimate the value of s_2 which gives the maximum value of the term $P(s_i \mid s_{i-1}, \omega_{i-1}, \omega_i, k, f_i, \boldsymbol{Y})$, here ($i = 2$). This process then continues until the end of the word ($i = n$).

3.4 Step 4

After we have estimated the parameter values of $\boldsymbol{S}$ and $\boldsymbol{F}$ from Step 1 to Step 3 for each dictionary word $\boldsymbol{\Omega}$ in each style k, we generate word image $\boldsymbol{W}$ with the estimated parameters and calculate the city block distance value between the input image $\boldsymbol{Y}$ and the generated image $\boldsymbol{W}$. Finally, we choose the word image with minimum distance value as the recognition result.

3.5 Making Character Templates

In this section, we describe how to make template images. Since the image to be recognized is a binary image in this paper, training data are also binary images. To make a template of each category in each font, first, we make gray scale images by summing all the training data for each category in each font. Then, we make two different binary images by binarizing the summed image with two different thresholds. We set three types of font and two types of binarization level for each font. Thus, these templates allow us to estimate the six types of style k for the input image.

Figure 6 shows examples for the template. The first pattern shows the summed image in three types of typewriter, the second pattern shows the binarized image with a high threshold value and the third with a low threshold value.

When we make summed images, the centre of gravity for each training data ample is located at the same position. Each threshold value is calculated with the maximum pixel value of the summed image (p_{max}). The high threshold value is calculated as $0.7 \times p_{max}$, and the low as $0.3 \times p_{max}$.

The number of images for each category in each font is not the same. For example, the number of "a" image is about 100, but in the case of "q" only about 20.

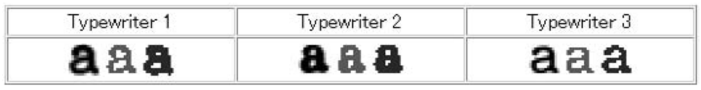


Fig. 6. Example templates

4 Evaluation

We evaluated the effectiveness of the proposed method using a word dictionary that consists of about 28,000 genus and species names and includes all the words in the test images. The number of images evaluated was 4468. The recognition rates with the proposed method and with a method proposed by Lucas (evaluated on the same database)[3] are shown in table 1 below. We evaluated with three types of parameter sets where the area to be searched in step2 and step3 is ±2 pixels horizontally and ±2 vertically as set1, ±2 pixels horizontally and ±4 pixels vertically as set2 and ±3 horizontally and ±4 vertically as set3.

Table 1. Recognition Rate

	Conventional	Set1(±2 × ±2)	Set2(±2 × ±4)	Set3(±3 × ±4)
1st	89.4%	99.5%	99.7%	99.8%
10-best	94.6%	99.8%	99.9%	99.9%

Example images that are correctly recognized by the proposed method (Set3) are shown in Figure 7.

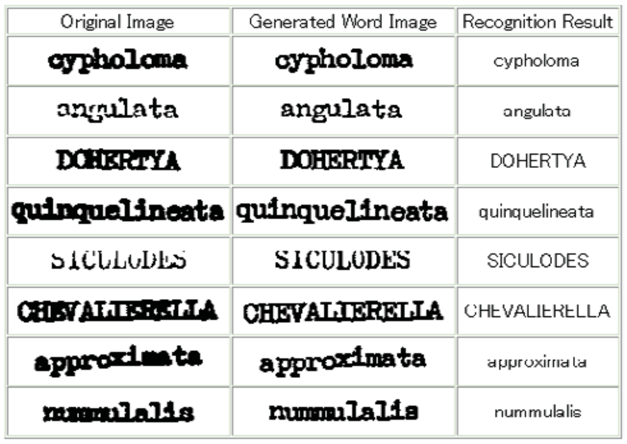
As may be seen in Figure 7, this method can correctly recognize word images including partly faded or heavily touching characters. The proposed method can also adapt to the images such as “approximata” in which characters are not arranged in a straight line.

Example images that are not correctly recognized by the proposed method are shown in Figure 8.

In the “ACHROIA” and “unicolor” case, the local minimum in step2 and step3 causes the error. In the “demotellus” case, the second character “e” is heavily touched by the next character “m”, so it is difficult to recognize correctly. In the “lutulentalis” case, the noise causes the error.

We also evaluated the non-holistic method where step 4 is not used for the recognition process. The score of the word C is calculated by using the equation below.

$$C = \sum_{i=1}^{n} \{S(\boldsymbol{Y}, s_i, f_i) - B\} \tag{4}$$



Original Image	Generated Word Image	Recognition Result
cypholoma	cypholoma	cypholoma
angulata	angulata	angulata
DOHERTYA	DOHERTYA	DOHERTYA
quinquelineata	quinquelineata	quinquelineata
SICULODES	SICULODES	SICULODES
CHEVALIERELLA	CHEVALIERELLA	CHEVALIERELLA
approximata	approximata	approximata
nummulalis	nummulalis	nummulalis

Fig. 7. Example images (correctly recognized)

Here, $S(\boldsymbol{Y}, s_i, f_i)$ is the city block distance value between a template f_i and a part image extracted from the input image $\boldsymbol{Y}$ by s_i , and B a bonus. We evaluated the recognition rate with various values of B. The relationship between the recognition rate and the values of B is shown in figure 9.

Results show that the recognition rate is maximum when the value of B is set at 50 in Set2. The recognition rate is 98.8%. In the case of Set3, the recognition rate is maximum (99.0%) when the value of B is set at 40. The recognition rate of the non-holistic method is lower than the proposed method, but still higher than the conventional method. The results show the importance of segmentation. In the conventional method, the arrangement of characters is assumed as all capital case characters. On the other hand, both in the non-holistic method and the

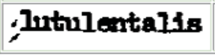
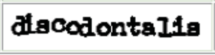
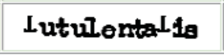
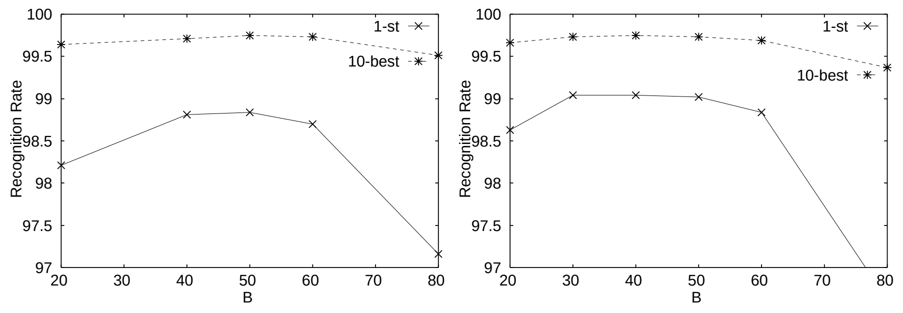
Original Image	Generated Word Image (1st rank)	Generated Word Image (Correct Script)	Correct Script	Recognition Result
			ACHROIA	SUFETULA
			unicolor	tricolor
			demotellus	damotellus
			lutulentalis	discodontalis

Fig. 8. Example images (erroneously recognized)

proposed method, the arrangement of the characters is not fixed. This causes the difference in the recognition rate between the conventional method and the non-holistic method.

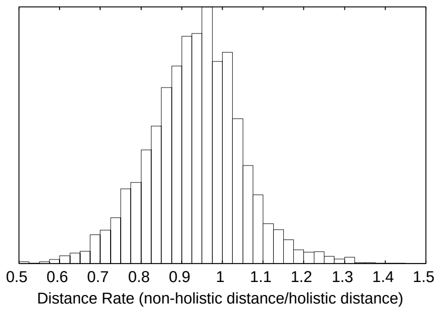


(b) Set2 ($\pm 2 \times \pm 4$) (c) Set3 ($\pm 3 \times \pm 4$)

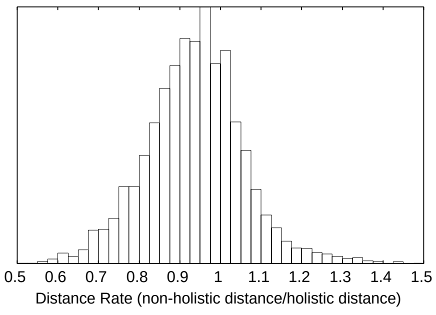
Fig. 9. Recognition Rate of non-holistic method on various value of B

When the individual character images are drawn on the input image arranged with enough gap between neighboring characters, the word score C calculated without bonus will be the same distance value as the proposed method. That is to say, the bonus means the approximated distance value that is calculated from the part of the input image that has no corresponding template. For example, when the input image consists of three characters but the word generated by the model consists of two characters, the last character's part should be calculated as a distance value. So the bonus B can be considered as a prefixed distance value corresponding to the third character's images.

We compared the distance value of the correct script obtained by the proposed method and the non-holistic method. Here, the distance value with the non-holistic method is the word score C calculated without bonus. Figures 10(a) and (b) show the distribution of the distance value of the correct script obtained by the non-holistic method where each value is divided by the distance value of the correct script obtained by the proposed method. Figure 10(a) is the results for Set2 and (b) for Set3.



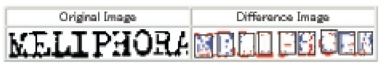
(a) Set2 ($\pm 2 \times \pm 4$)



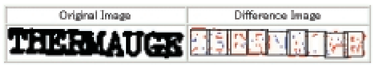
(b) Set3 ($\pm 3 \times \pm 4$)

Fig. 10. Distribution of the distance value of correct script obtained by non-holistic method

As may be seen in figures 10(a) and (b), these wide distributions mean that we have error accumulation during the process in the non-holistic method. The small errors will accumulate in the non-holistic method because the non-holistic method can't cover the whole word image but only the part image extracted by S (see figure 11(a)) and also can't eliminate the black pixels intruding in the recognition window from neighboring characters (figure 11(b)). Figure 11 (a) and (b) shows the original image and the difference image. The rectangles in the difference image are the recognition window for each character. In the case of (a), the pixels outside the rectangles are not used in calculating the distance value. On the other hand, in the case of (b), the rectangles in the difference image are overlapping each other. We may assume that these errors have expectation value of zero and enough small variance. However, our results show that the assumption above is not reasonable and the accumulated error damages the recognition rate in the non-holistic method. In the proposed method, we can expect that no such errors will accumulate.



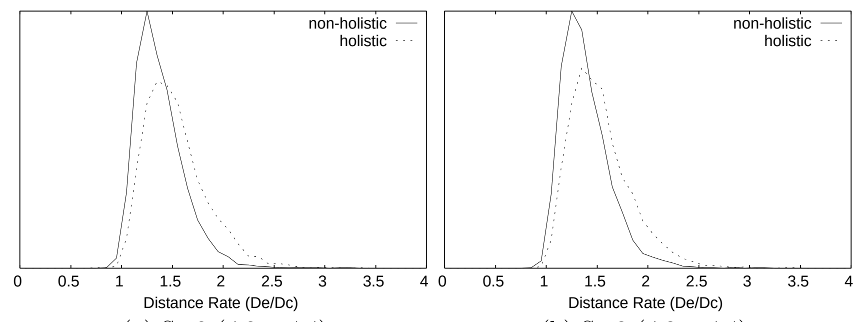
(a)



(b)

Fig. 11. Example Images

Next, we compared the distance value of the correct script (Dc) and the minimum distance value of the erroneous script (De). For example, when the first recognition candidate is correct, the distance value of the second candidate is the minimum value of erroneous script. To calculate the De and Dc with the non-holistic method, first, we calculate the word score C with equation (4), and then add $B \times N_c$ to each word score. Here, N_c is the number of characters in the correct script[1]. Figures 12(a) and (b) show the distribution of the erroneous script on De/Dc. Figure 12(a) is the results of Set2 with the value of $B = 50$ and (b) Set3 with $B = 40$.



(a) Set2 ($\pm 2 \times \pm 4$) (b) Set3 ($\pm 3 \times \pm 4$)

Fig. 12. Distribution of the distance value of erroneous script

As in figure 12(a) and (b), the proposed method can separate the correct candidate from erroneous candidates better than the non-holistic method.

These results show that the difference between the conventional method and the non-holistic method is caused by the difference of segmentation, and the difference between the non-holistic method and the proposed method is caused by the difference of the error accumulation.

5 Conclusion

We have proposed a new top-down likelihood word image generation model for word recognition. We have defined a word image generation model and word recognition problem with the model proposed here. The word recognition algorithm is defined as a parameterised search problem.

The difference between a non-holistic method and the proposed method is shown by the evaluation. The small errors accumulate in non-holistic methods during the process carried out. We confirmed that such an accumulated error seriously damages the recognition rate in the non-holistic method. Results show that a recognition rate of 99.8% was obtained in the proposed method.

Future work aims to extend this model for non dictionary applications using a character's bi-gram, where it is very important to estimate a likelihood value for each term accurately.

[1] We add $B \times N_c$ even in the case that the number of the characters in the erroneous script is different to the number in correct script. This process doesn't change the rank order in the recognition results obtained by the non-holistic method, but just shifts all the word score. This process roughly normalizes the word score C of the correct script. Then it will be easy to compare the separation of the correct candidate from erroneous candidate between the two.

Acknowledgements. We are grateful to the UK Natural History Museum for kindly providing the archive card database under an EPSRC/BBSRC Bioinformatics research contract 84/BIO11933.

References

1. A. C. Downton et al.: "Constructing Web-Based Legacy Index Card Archives – Architectual Design Issues and Initial Data Acquisition", ICDAR'01, 2001.
2. M. Sawaki, N. Hagita: "Text-line Extraction and Character Recognition of Document Headlines with Graphical Designs using Complementary Similarity Measure", IEEE trans. PAMI, vol. 20, No. 10, pp 1103-1109, 1998.
3. S. M. Lucas et al.: "Robust Word Recognition for Museum Archive Card Indexing", ICDAR'01, 2001.
4. M. A. Ozdil, F.T.Y. Vural: "Optical Character Recognition without Segmentation", ICDAR'97, 1997
5. S. Madhvanath, E. Kleinberg, V. Govindaraju: "Holistic Verification of Handwritten Phrases", IEEE trans. PAMI, vol. 21, No. 12, pp 1344-1356, 1999.
6. S. Madhvanath, V. Govindaraju: "The Role of Holistic Paradigms in Handwritten Word Recognition", IEEE trans. PAMI, vol. 23, No. 2, pp. 149-164, 2001.
7. Y. Lu, C.L. Tan, W.Huang, L.Fan:"An Approach to Word Image Matching Based on Weighted Hausdorff Distance", ICDAR'01, 2001.
8. R. Plamondon, S.N. Srihari: "On-Line and Off-Line Handwriting Recognition: A Comprehensive Survey", IEEE trans. PAMI, vol. 22, No. 1, pp 63-84, 2000.
9. E. Lethelier, M. Leroux, M. Gilloux: "An Automatic Reading System for Handwritten Numeral Amounts on French Checks", ICDAR'95, 1995
10. M. Gilloux, B.Lemaré, M. Leroux: "A Hybrid Radial Basis Function Network/Hidden Markov Model Handwritten Recognition System", ICDAR'95, 1995
11. I. Bazzi, C. LaPre, J. Makhoul, C. Raphael, R. Schwartz: " Omnifont and Unlimited-Vocabulary OCR for English and Arabic",ICDAR'97,1997.
12. P. Sarkar, G. Nagy: "Style-consistency in isogenous patterns", ICDAR'01, 2001.