An *A Posteriori* Probability Calculation Method for Analytic Word Recognition Applicable to Address Recognition

Tomoyuki Hamamura, Takuma Akagi, Bunpei Irie
TOSHIBA Corporation
tomoyuki.hamamura@toshiba.co.jp

Abstract

In this paper, we propose a novel calculation method of an "a posteriori" probability for analytic word recognition. The method is suitable for address recognition tasks. Our previous method needs calculation over all words in a lexicon, while the proposed method only needs calculation on the concerned word. In the address recognition task, lexicon size becomes very large and only a part of it can be handled. Hence, the previous method cannot be used, and the proposed method is convenient. Experimental results show that the proposed method guarantees high precision of probability calculation.

1. Introduction

Word recognition algorithms are classified into two major groups: an "analytic" approach and a "holistic" approach [9]. In an analytic approach, segmentation is performed to make character hypotheses and character recognition module is used to process each hypothesis. In a holistic approach, word recognition is performed without character recognition. Hidden Markov Model (HMM) is mainly used in this approach.

Although an analytic approach is an old, conventional one, it is still an effective technique. Kimura et al. took the first place among single algorithms in ICDAR 2007 Arabic handwriting recognition competition [8]. Their system is the only one using an analytic approach [1]. Koerich et al. made a great improvement of the recognition rate adding a postprocess of character recognition after their HMM based algorithm [6]. In general, combination of both approaches leads to high performance [9].

The goal of a word recognition task in an analytic approach is to choose the word in a lexicon that best matches a consistent combination of the character hypotheses (a consistent combination of the character hypotheses is referred to as "path" below). To achieve this, the most common way is to calculate an evaluation value for each pair of the path and the lexicon word, and choose the pair with the highest evaluation value. Such an approach, in which the lexicon word and the path (= segmentation) is determined simultaneously, is referred to as a "lexicon driven" approach.

Here, a calculation rule of an evaluation value (= evaluation function) becomes key issues of a recognition method. In previous works, empirical functions were mainly employed for this purpose, compared to holistic approaches, which mainly employs statistical methods like HMM. The mean of character recognition scores (similarity, confidence value, reliability, distance, cost, etc.) was the most common evaluation function [6, 3, 7, 5]. By Bayes' decision rule, the best evaluation function is an *a posteriori* probability of the word in the lexicon. In the case that segmentation is known (e.g. postal code in pre-printed boxes), it has been known for several decades that the *a posteriori* probability can be calculated using the product of a likelihood of each character [1]. HMM is also based on the same principle. But, in the case that segmentation is unknown and a word can be in various length, it had been difficult to calculate the *a posteriori* probability. In our previous work, we have succeeded in calculating the *a posteriori* probability approximately in such a case with significant improvement in word recognition performance [4].

However, this calculation method uses all words in the lexicon in order to calculate an *a posteriori* probability of one word. This has become a problem in handling a huge size lexicon such as an address recognition task [10, 2]. Therefore, a method to calculate the *a posteriori* probability without using all words has been anticipated.

In this paper, we propose a novel calculation method which can calculate the *a posteriori* probability using only one word. This calculation method is derived by finding a way of approximation. An experiment in word recognition shows that the proposed method is far superior to the widely used evaluation function, although it is slightly inferior to our previous method. Organization of this paper is as follows. Section 2 and 3 will describe our previous method. We will give a detailed formulation of the problem (section 2) and derivation of the formulae (section 3), described briefly in [4]. Section 4 will explain the problem of the previous method in an address recognition task, and derive the proposed method in section 5. Section 6 is about an experiment and results. Summary of the work and some concluding remarks are presented in Section 7.

[1] Kimura et al. was the second best, while the best one was the system combining three different algorithms.

978-0-7695-3725-2/09 $25.00 © 2009 IEEE
DOI 10.1109/ICDAR.2009.240

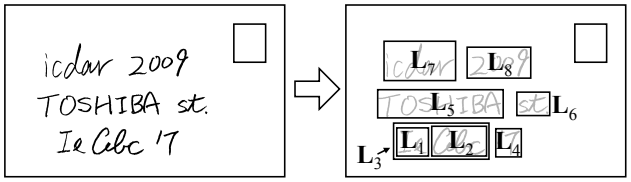

Figure 1. Multiple word hypotheses

2. A problem definition

Assume that the location of a word in the lexicon to be unknown, and multiple word hypotheses have been generated in advance. Fig.1 shows an example of word hypotheses on a mail image, where the address "icdar 2009 TOSHIBA st Ie Abc 7" is written. Our goal is to determine which word in the lexicon is written in the image. Let us consider that there is only one hypothesis in which a word in the lexicon is written. For example, consider a problem of reading a city name in a mail. This situation can also be regarded as a "word spotting" problem.

Since a boundary between characters is unknown, we extract many segmentation points as a candidate of a boundary in advance so that correct boundary points are all included in the candidates, which is referred to as "over-segmentation". Fig.2(a) shows an example of segmentation points. The line image corresponds to the bottom line of Fig.1. "↑" means a segmentation point. Character hypotheses are generated according to the segmentation points. The structure of the character hypotheses is in the form of "lattice". Fig.2(b) is an illustration of the lattice generated from Fig.2(a). $\mathbf{L}_1$ to $\mathbf{L}_4$ denotes word hypotheses corresponding to Fig.1.

One segmentation hypothesis corresponds to one "path" on the lattice. Fig.2(c) shows examples of a path. If all correct boundary points are included in segmentation points, one of the paths necessarily corresponds to the correct segmentation. In Fig.2(c), $\mathbf{f_1}$ is a correct path, and all character hypotheses on the path correspond to characters in the line image.

Character recognition is performed to each character hypothesis. Fig.2(d) illustrates character recognition results. In general, a recognition result includes multiple character code candidates and their scores (the score is called similarity, confidence, reliability, distance, etc), but as an example, only the top candidate and its score (similarity) is shown in the figure. Our goal is to choose a path, a lexicon word, and a word hypothesis simultaneously using the above information [2].

3. The previous method

According to Bayes' decision rule, choosing a word from the lexicon which maximizes the *a posteriori* probability minimizes the error rate. Therefore, we try to calculate the

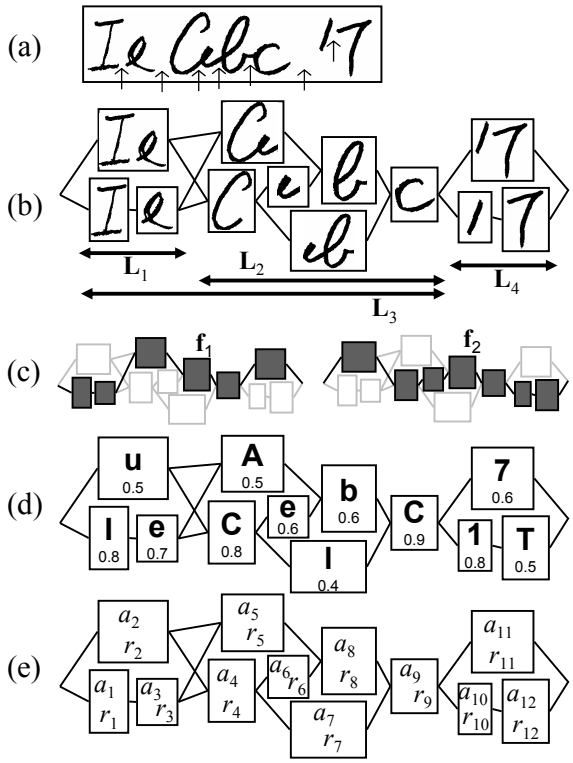

Figure 2. Character hypotheses

a posteriori probability $P(w_i|\mathbf{x})$ approximately, where w_i denotes i-th word in the lexicon and $\mathbf{x}$ denotes the "whole" information of the image, which is described in detail below [3].

Then, let us define several notations. a_i denotes the i-th character hypothesis. r_i denotes the character recognition result of i-th character hypothesis a_i (see Fig.2(e)). Here, the character recognition result r_i can mean any information given by character recognition process of a_i. For example, it can mean the only top candidate of recognition result (Fig.2(d)), or all the candidates and their reliability values. A=$\{a_1, a_2, ..., a_m\}$ denotes the set of all the character hypotheses, where m is the number of the character hypotheses. $\mathbf{r}$=$(r_1, r_2, ..., r_m)$ denotes the character recognition result of all the character hypotheses. $\mathbf{S}$ denotes the compilation of all information except the character recognition result $\mathbf{r}$, such as the arrangement of the character hypotheses, size of each character hypothesis, the number of the character hypotheses, etc. $\mathbf{x}$ mentioned above is defined as $\mathbf{x}$=$(\mathbf{r}, \mathbf{S})$. $\mathbf{f}_p$ denotes the p-th path in the lattice of character hypotheses, which is defined as $\mathbf{f}_p$=$(a_{f_1^p}, a_{f_2^p}, ...), a_{f_i^p} \in A, 1 \leq f_i^p \leq m$. $\mathcal{F} = \{\mathbf{f}_p\}$, p=1,2,... denotes the set of all the paths in the lattice. $\mathbf{L}_l$ denotes the l-th word hypothesis. $\mathcal{L}$=$\{\mathbf{L}_l\}$ denotes the set of all the word hypotheses. When the recognition object is split into multiple lines (e.g. Fig.1), it is treated as in a single line virtually by concatenation.

[2] Utilization of the information other than character recognition results (e.g. geometric relationship of characters) will be our future work, which will improve the performance.

[3] The optimal path and word hypothesis will also be found, although only the identification of the word in the lexicon is explained.

By Bayes' theorem,

$$P(w_i|\mathbf{x}) = P(\mathbf{x},w_i)/P(\mathbf{x}) \quad (1)$$
$$= P(\mathbf{x},w_i)/\textstyle\sum_k P(\mathbf{x},w_k). \quad (2)$$

Since the numerator and the denominator have similar terms in Eq.(2), we transform the numerator below.

$$P(\mathbf{x},w_i) = P(\mathbf{r},\mathbf{S},w_i) = \sum_{\mathbf{f}_p\in\mathcal{F},\mathbf{L}_l\in\mathcal{L}} P(\mathbf{r},\mathbf{S},\mathbf{f}_p,\mathbf{L}_l,w_i)$$
$$\approx \max_{\mathbf{f}_p\in\mathcal{F},\mathbf{L}_l\in\mathcal{L}} P(\mathbf{r},\mathbf{S},\mathbf{f}_p,\mathbf{L}_l,w_i) \quad (3)$$
$$= \max_{\mathbf{f}_p\in\mathcal{F},\mathbf{L}_l\in\mathcal{L}} \Big\{P(\mathbf{r}|\mathbf{S},\mathbf{f}_p,\mathbf{L}_l,w_i)P(\mathbf{S},\mathbf{f}_p,\mathbf{L}_l|w_i)P(w_i)\Big\} \quad (4)$$

The approximation in Eq.(3) means that probabilities is negligible except for the maximum one.

Next, we will employ the following approximation:

$$P(\mathbf{S},\mathbf{f}_p,\mathbf{L}_l|w_i) \approx \begin{cases} K, & if\ \ len(w_i)=v_{l,p}-u_{l,p}+1 \\ 0, & otherwise \end{cases} \quad (5)$$

where K is a constant irrelevant to p, l, and i, $len(w_i)$ denotes the number of character codes in the word w_i, and $u_{l,p}$ and $v_{l,p}$ denotes the index of the "boundary" character hypothesis in the path $\mathbf{f}_p$, that is, the character hypothesis $a_{f_j^p}$ in the path $\mathbf{f}_p$ is contained in the word hypothesis $\mathbf{L}_l$ if and only if $u_{l,p}\le j\le v_{l,p}$. The approximation in Eq.(5) is based on the model that a probability of an arbitrary word matching with an arbitrary path is equally likely, if the length of the word and the length of the path are equal, without knowledge of the character recognition results $\mathbf{r}$. A more suitable model of $P(\mathbf{S},\mathbf{f}_p,\mathbf{L}_l|w_i)$ will make a better approximation, if we can find one.

By using Eq.(5), Eq.(4) is transformed as follows: (p and l are abbreviated from the following equation.)

$$P(\mathbf{x},w_i) \approx KP(w_i) \max_{\mathbf{f}\in\mathcal{F},\mathbf{L}\in\mathcal{L}} P(\mathbf{r}|\mathbf{S},\mathbf{f},\mathbf{L},w_i) \quad (6)$$
$$\approx K'\max_{\substack{\mathbf{f}\in\mathcal{F}\\ \mathbf{L}\in\mathcal{L}}} \Big\{\prod_{u\le j\le v} P(r_{f_j}|c_{i,j-u+1}) \prod_{j<u,v<j} P(r_{f_j}|C) \prod_{a_j\in D} P(r_j)\Big\} \quad (7)$$
$$= K'\Big\{\prod_{a_j\in A} P(r_j)\Big\} \max_{\substack{\mathbf{f}\in\mathcal{F}\\ \mathbf{L}\in\mathcal{L}}} \Big\{\prod_{u\le j\le v} \frac{P(r_{f_j}|c_{i,j-u+1})}{P(r_{f_j})} \prod_{j<u,v<j} \frac{P(r_{f_j}|C)}{P(r_{f_j})}\Big\} \quad (8)$$
$$= K''\max_{\mathbf{L}\in\mathcal{L}} \Big[\max_{\mathbf{f}\in\mathcal{F}} \Big\{\prod_{u\le j\le v} R(r_{f_j}|c_{i,j-u+1}) \prod_{j<u,v<j} R(r_{f_j}|C)\Big\}\Big] \quad (9)$$
$$= K''\max_{\mathbf{L}\in\mathcal{L}} \Big[\max_{\mathbf{n}\in\mathcal{N}} \Big\{\prod_j R(r_{n_j}|c_{ij})\Big\} \max_{\mathbf{m}\in\mathcal{M}} \Big\{\prod_j R(r_{m_j}|C)\Big\}\Big] \quad (10)$$
$$= K''\max_{\mathbf{f}\in\mathcal{F}} \Big\{\prod_j R(r_{f_j}|C)\Big\} \max_{\mathbf{L}\in\mathcal{L}} \Big[\frac{\max_{\mathbf{n}\in\mathcal{N}} \big\{\prod_j R(r_{n_j}|c_{ij})\big\}}{\max_{\mathbf{n}\in\mathcal{N}} \big\{\prod_j R(r_{n_j}|C)\big\}}\Big] \quad (11)$$

In Eq.(7), K' is defined as $K'=KP(w_i)$. $c_{i,k}$ denotes the k-th character code in the word w_i (e.g. if w_i is "SPAIN", $c_{i,3}$ is "A"). C denotes an arbitrary character code, that is, C means a correctly segmented character hypothesis, because a correctly segmented character hypothesis must correspond to some character code. Hence, $P(r_k|C)$ means the probability of the character recognition result of the character hypothesis a_k being r_k on the condition that a_k is correctly segmented. D_p denotes the set of character hypotheses which do not belong to the path $\mathbf{f}_p$ (e.g. in Fig.2(c), D_2 is $\{a_2,a_5,a_6,a_8\}$, that are not on the path $\mathbf{f}_2$). The approximation in Eq.(7) is based on the assumption that the character recognition results of the character hypotheses are independent of each other is employed. Factors of the first $\prod$ correspond to character hypotheses that are on the path $\mathbf{f}_p$ and inside the word hypothesis $\mathbf{L}_l$ (e.g. a_4,a_6,a_8,a_9, if $\mathbf{f}_p=\mathbf{f}_2$ and $\mathbf{L}_l=\mathbf{L}_2$ in Fig.2). Factors of the second $\prod$ correspond to those on $\mathbf{f}_p$ and outside $\mathbf{L}_l$ (e.g. a_2,a_{10},a_{12}). Factors of the third $\prod$ correspond to those that are not on $\mathbf{f}_p$ (e.g. a_1,a_3,a_5,a_7,a_{11}).

In Eq.(8), exploiting the fact that the product over all the character hypotheses $\prod_{a_j\in A}P(r_j)$ is irrelevant of $\mathbf{f}_p$ and $\mathbf{L}_l$, the inside of max is divided by this product. Therefore, the third $\prod$ in Eq.(7) can be canceled.

In Eq.(9), we define $K''=K'\big\{\prod_{a_j\in A}P(r_j)\big\}$ and $R(a|b)=\frac{P(a|b)}{P(a)}$. First $\prod$ in Eq.(9) is inside the word hypothesis $\mathbf{L}_l$, and second $\prod$ is outside $\mathbf{L}_l$. Here, we assume that there is no character hypothesis spanning a boundary of a word hypothesis. Then, the inside and the outside of $\mathbf{L}_l$ can be maximized independently, which leads to Eq.(10).

In Eq.(10), $\mathbf{n}^l_{p'}$ denotes the p'-th path inside the word hypothesis $\mathbf{L}_l$, which is defined as $\mathbf{n}^l_{p'}=(a_{n_1^{lp'}},a_{n_2^{lp'}},...)$, $a_{n_i^{lp'}}\in A$ (one example of $\mathbf{n}^2_{p'}$ is (a_4,a_7,a_9) in Fig.2(e)). $\mathcal{N}^l=\{\mathbf{n}^l_{p'}\}$ (p'=1, 2, ...) denotes the set of all the paths inside the word hypothesis $\mathbf{L}_l$. $\mathbf{m}^l_{p''}$ denotes the p''-th "concatenated path" outside the word hypothesis $\mathbf{L}_l$. The "concatenated path" means a concatenation of two paths, one is a path in the left side of $\mathbf{L}_l$, the other is in the right. $\mathbf{m}^l_{p''}$ is defined as $\mathbf{m}^l_{p''}=(a_{m_1^{lp''}},a_{m_2^{lp''}},...)$, $a_{m_i^{lp''}}\in A$ (one example of $\mathbf{m}^2_{p''}$ is (a_1,a_3,a_{10},a_{12})). $\mathcal{M}^l=\{\mathbf{m}^l_{p''}\}$ (p''=1, 2, ...) denotes the set of all the "concatenated paths" outside the word hypothesis $\mathbf{L}_l$.

For deriving Eq.(11), following equation is exploited:

$$\max_{\mathbf{f}\in\mathcal{F}}\Big\{\prod_j R(r_{f_j}|C)\Big\} = \max_{\mathbf{n}\in\mathcal{N}}\Big\{\prod_j R(r_{n_j}|C)\Big\} \max_{\mathbf{m}\in\mathcal{M}}\Big\{\prod_j R(r_{m_j}|C)\Big\}.$$

Since $\max_{\mathbf{f}\in\mathcal{F}}\Big\{\prod_j R(r_{f_j}|C)\Big\}$ is irrelevant of $\mathbf{L}_l$, the inside of $\max_{\mathbf{L}\in\mathcal{L}}$ in Eq.(10) can be divided by it, and then $\max_{\mathbf{m}\in\mathcal{M}}$ can be canceled.

Substituting Eq.(11) into Eq.(2) gives

$$P(w_i|\mathbf{x}) \approx \frac{value(w_i)}{\sum_k value(w_k)}, \quad (12)$$

where $value()$ is defined as:

$$value(w_i) = P(w_i) \max_{\mathbf{L} \in \mathcal{L}} \left[\frac{\max_{\mathbf{n} \in \mathcal{N}} \left\{ \prod_j R(r_{n_j}|c_{ij}) \right\}}{\max_{\mathbf{n} \in \mathcal{N}} \left\{ \prod_j R(r_{n_j}|C) \right\}} \right]. \quad (13)$$

where

$$R(r_{n_j}|c_{ij}) = \frac{P(r_{n_j}|c_{ij})}{P(r_{n_j})}, R(r_{n_j}|C) = \frac{P(r_{n_j}|C)}{P(r_{n_j})}. \quad (14)$$

A posteriori probabilities can be calculated approximately using Eq.(12)-(14). For example in Eq.(14), $P(r_{n_j}|c_{ij})$ equals the ratio of patterns with the recognition result "$\mathbf{I}, 0.8$" in the set of "A", if c_{ij} = "A" and $r_{n_j} = r_1$ in Fig.2(d),(e). You must find some way when you have not enough patterns. For example, it may be helpful to substitute $P(\text{"not}\mathbf{A}, 0.8\text{"}|\text{"A"})$ for $P(\text{"}\mathbf{I}, 0.8\text{"}|\text{"A"})$, where "not$\mathbf{A}$" means the top candidate is not "$\mathbf{A}$". You may use the code of the top candidate only, or use the second and other candidates, with/without the scores as r, instead of the code and the score of the top candidate, in this example. Likewise, $P(r_{n_j}|C)$ can be calculated from the set of correctly segmented characters. $P(r_{n_j})$ can be calculated from the set of all the character hypotheses. Dynamic programming technique can be employed for finding the optimal path $\mathbf{n}$ in Eq.(13). You may assume $P(w_i)$ in Eq.(13), an *a priori* probability of w_i to be all equal, or you may estimate them by training data.

4. Problems of the previous method

The previous method requires processing all the words in the lexicon for computing the *a posteriori* probability of a single word. $value(w_k)$ for all the words must be calculated in the denominator of Eq.(12). This causes problems in address recognition. Since the lexicon used in address recognition is very large. The processing of the whole data estimated from millions to tens of millions is unrealistic for delivery point identification. The lexicon is usually generated dynamically in address recognition. For example, a postcode is recognized beforehand to shrink the street lexicon [10], city name is recognized to shrink the town lexicon [2]. Eq.(12) can easily be calculated for the shrunk lexicon formally. However, there comes up an error, since not all the data is used in the denominator. Especially, the error increases when pruning is unsuccessful. Because the total sum is normalized to 1 in Eq.(12), which prevents the *a posteriori* probability $P(w_i|\mathbf{x})$ from becoming a small value even when the true answer is not included in the lexicon. Consequently, unsuccessful pruning cannot be detected.

The *a posteriori* probability will be small as unsuccessful pruning, if it can be calculated precisely without using all the data. It also enables the detection of pruning failure. In this case, regeneration of the lexicon is possible by backtracking. For the reason mentioned above, the calculation method of the *a posteriori* probability without using all the words in the lexicon is needed. In the next section, we demonstrate the numerator $value(w_i)$ in Eq.(12) alone can be regarded as an approximation of the *a posteriori* probability.

5. The proposed method

In previous derivation, owing to starting from Eq.(2), expensive calculation for all words in the lexicon is required in Eq.(12). Therefore, we try to start from Eq.(1). The denominator of Eq.(1) is transformed as follows: (p is abbreviated from following equation.)

$$P(\mathbf{x}) = P(\mathbf{r}, \mathbf{S}) = \sum_{\mathbf{f} \in \mathcal{F}} P(\mathbf{r}, \mathbf{S}, \mathbf{f}) \approx \max_{\mathbf{f} \in \mathcal{F}} P(\mathbf{r}, \mathbf{S}, \mathbf{f}) \quad (15)$$

$$= \max_{\mathbf{f} \in \mathcal{F}} \Big\{ P(\mathbf{r}|\mathbf{S}, \mathbf{f}) P(\mathbf{S}, \mathbf{f}) \Big\} \approx K \max_{\mathbf{f} \in \mathcal{F}} \Big\{ P(\mathbf{r}|\mathbf{S}, \mathbf{f}) \Big\} \quad (16)$$

$$\approx K \max_{\mathbf{f} \in \mathcal{F}} \Big\{ \prod_j P(r_{f_j}|C) \prod_{a_j \in D} P(r_j) \Big\} \quad (17)$$

$$= K \Big\{ \prod_{a_j \in A} P(r_j) \Big\} \max_{\mathbf{f} \in \mathcal{F}} \Big\{ \prod_j R(r_{f_j}|C) \Big\} \quad (18)$$

where we use same notation as the one in previous section. Above conversion resembles that of Eq.(3) to (11), based on the same idea.

The approximation of $P(\mathbf{S}, \mathbf{f}_p) \approx K$ is used in Eq.(16). No enumeration is needed as in Eq.(5), since w_i is not given. This approximation is based on the assumption that the probability of every path is equal, as in Eq.(5). A better model may improve the performance, since the approximation of using the same value K as in Eq.(5) is rather rough.

In Eq.(17), the approximation that the character recognition results of the character hypotheses are independent of each other is employed, the same as Eq.(7). In Eq.(18), the inside of max in Eq.(17) is divided by $\prod_{a_j \in A} P(r_j)$, the same as Eq.(8).

Substituting Eq.(11) and Eq.(18) into Eq.(1) gives,

$$P(w_i|\mathbf{x}) \approx value(w_i). \quad (19)$$

Eq.(19) shows that the numerator of Eq.(12) itself can be used as an approximation of the *a posteriori* probability. Thus, Eq.(19) is a solution for the problem of the previous method pointed out in section 4.

The great difference between the proposed method and the previous one is the treatment of the denominator $P(\mathbf{x})$ of Eq.(1). The previous method can be seen as calculating $P(\mathbf{x})$ using all the words in the lexicon. On the other hand in the proposed method, $P(\mathbf{x})$ is calculated using an arbitrary character code C in Eq.(17). In another word, the proposed method is approximating the lexicon by "arbitrary string".

6. Experiment

We conducted an experiment of recognizing city name in the mail images to evaluate effectiveness of the proposed method. We used 811 real mail images of a European country for the test. Fig.3 shows two examples of the whole

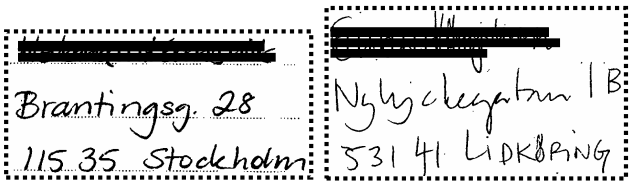

Figure 3. Examples of the whole address area

address area of a real image (parts of images are covered with black rectangles due to protection of personal information). Those images without the correct word hypothesis due to the upstream processes (address area detection, word segmentation, etc.), were excluded from the test set.

There were 1646 city names registered on the lexicon. Those images with unregistered city name were excluded from the test set. We employed a subspace method [11] for the character recognition classifier. The templates of the classifier were designed using a different image database. $R(r_k|c_{i,j})$ and $R(r_k|C)$ were calculated using yet another independent database. A contour analysis based character segmentation algorithm was employed.

Table 1 shows the recognition rate for three different evaluation functions: (A) a mean of similarities between a character hypothesis and character code in a word; (B) the previous method described in section 3; (C) the proposed method. The recognition rate of C is 22.8% higher than A. The rate of C is theoretically identical with B on the condition of no rejection, because Eq.(12) and Eq.(19) are maximized simultaneously when $value(w_i)$ is maximized.

Fig.4 shows the error-rejection curve of the three evaluation functions. Here, (error rate) = (# of error) / (# of all images) and (rejection rate) = (# of rejection) / (# of all images). For A, each point on the curve was determined by optimizing the reject threshold th_1 for the similarity of the first candidate and the reject threshold th_2 for a difference of the similarity of the first and the second candidate. For B and C, a threshold for the *a posteriori* probability was varied to draw the curve. The x-intercepts of the curves are equal to the values in Table 1. The graph shows clearly that the proposed method C is far better than A, and is not so inferior to B.

7. Conclusion

In this paper, we proposed the novel *a posteriori* probability calculation method for analytic word recognition. Even if we calculate the *a posteriori* probability of only one word, our previous method needs calculation over all words in the lexicon, while the proposed method needs the concerned word only. Therefore, the proposed method is suitable for address recognition which uses the huge size lexicon. The experiment on real mail images indicated the proposed method was superior to widely used evaluation function, although it was slightly inferior to our previous method.

Table 1. Recognition rate

A	B	C
48.3%	71.1%	71.1%

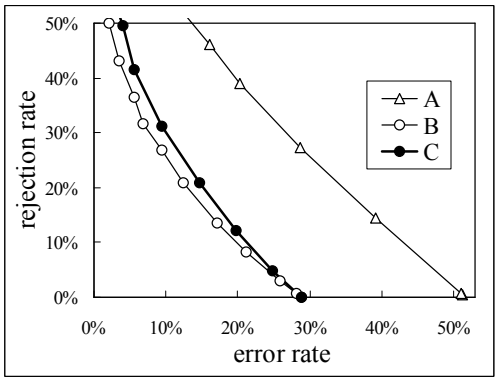

Figure 4. Error-rejection curve

References

[1] K. Abe, K. Hatano, and T. Fukumura. Performance evaluation of character recognition system with dictionary. *IEICE*, 52-C(6):305–312, Jun. 1969 (in Japanese).

[2] T. Akiyama, D. Nishiwaki, E. Ishidera, K. Kondoh, M. Hayashi, and T. Yamauchi. Alphabetical handwritten address recognition method allowing for no postal code and omission of several address elements. *Technical Report of IEICE*, PRMU2002-244:7–12, Mar. 2003 (in Japanese).

[3] P. D. Gader, M. Mohammed, and J.-H. Chiang. Handwritten word recognition with character and inter-character neural networks. *IEEE Trans. Systems, Man, and Cybernetics*, 27(1):158–164, Feb. 1997.

[4] T. Hamamura, T. Akagi, and B. Irie. An analytic word recognition algorithm using a posteriori probability. *Proc. 9th ICDAR*, 2:669–673, 2007.

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

[6] A. L. Koerich, R. Sabourin, and C. Y. Suen. Recognition and verification of unconstrained handwritten words. *IEEE Trans. PAMI*, 27(10):1509–1522, Oct. 2005.

[7] C.-L. Liu, M. Koga, and H. Fujisawa. Lexicon-driven segmentation and recognition of handwritten character strings for japanese address reading. *IEEE Trans. PAMI*, 24(11):1425–1437, Nov. 2002.

[8] V. Margner and H. E. Abed. Icdar 2007 arabic handwriting recognition competition. *Proc. 9th ICDAR*, pages 1274–1278, 2007.

[9] R. Plamondon and S. N. Srihari. On-line and off-line handwriting recognition: A comprehensive survey. *IEEE Trans. PAMI*, 22(1):63–84, Jan. 2000.

[10] S. N. Srihari. Handwritten address interpretation: A task of many pattern recognition problems. *Int'l J. Pattern Recognition and Artificial Intelligence*, 14(5):663–674, 2000.

[11] S. Watanabe and N. Pakvasa. Subspace method of pattern recognition. *Proc. 1st. IJCPR*, pages 25–32, 1973.