

**An On-line Handwritten Japanese Text
Recognition System Free from Line Direction and
Character Orientation Constraints**

文字方向および行方向に依存しないオンライン
日本語手書き枠なし文字列認識

Yuechan Hao(郝 月嬋)

Supervisor: Prof. Masaki Nakagawa

Department of Electronic and Information Engineering

Tokyo University of Agriculture and Technology

Abstract

This paper describes a recognition system for on-line handwritten Japanese text free from line direction and character orientation constraints.

Due to the increasing size of writing surface on a PDA, and the advent of a tablet PC or an electronic whiteboard, people can write text more freely as on a piece of paper. Thus, the demand to remove writing constraints from on-line handwriting recognition is getting higher. A new type of pen interfaces like E-pen and paper interface by Anotopen and paper are also raising this demand even higher. Such freely handwritten text brings new challenges to remove writing constraint from on-line handwriting recognition. Asian people whose languages are Chinese origin often write text horizontally, vertically or even slantingly in a mixed way.

The recognition system separates handwritten text of arbitrary character orientation and line direction into text line elements, estimates and normalizes character orientation and line direction, applies two-stage over-segmentation, constructs a segmentation-recognition candidate lattice and evaluates the likelihood of candidate segmentation-recognition paths by combining the scores of character recognition, geometric features and linguistic context.

Due to the text lines are free from line direction and character orientation, the unstable line direction and character orientation it is a challenging research work to segment text line correctly and segment characters correctly for every line direction before recognition as human for machine. To solve the problem, at text line segmentation step, we segment text lines compose of horizontal, vertical and slanted lines of text with arbitrary character orientation into text line elements. At over segmentation step, we decide segmentation points and non-segmentation points in quantized 4 directions using the two-stage classification scheme. And then evaluate the likelihood of candidate segmentation paths, train and decide the weight of each factor automatically by genetic algorithm. Finally optimal path can be found by the Viterbi search.

The results of experiments on text from the *HANDS-Kondate_t_bf-2001-11* database demonstrate significant improvements in the character recognition rate compared with the previous systems.

Its recognition rate on text of arbitrary character orientation and line direction is now comparable with that possible on horizontal text with normal character orientation. Moreover, its recognition speed and memory requirement do not limit the platforms or applications that employ the recognition system. This is a common research with the company iLabo, and as

product for customer.

There remain several works to improve the performance. We need to try other methods to get a more robust classification result, such as Convolutional Neural Network (CNN) architecture, which are able to effectively utilize the contextual information.

In chapter 1, we briefly describe the background and the objective of this study. Then, we introduce the organization of this thesis.

In chapter 2, we mainly give a survey on the state-of-the-art methods for on-line handwritten text recognition, and the orientation free handwritten text recognition.

In chapter 3, we briefly describe the character recognition system combining on-line and off-line character recognizers, for each candidate character pattern in the candidate lattice.

In chapter 4, we describe the recognition methods for character-orientation-free and line-direction-free on-line handwritten Japanese text recognition.

In chapter 5, we describe the linguistic context and geometric context for the path evaluation criterion to improve the text recognition accuracy.

In chapter 6, we describe the experiments. We compare the results of the proposed methods with the Onuma et al. system [11], and give some analyses of recognition performance.

In chapter 7, we conclude this research and give several directions for the future work.

論文要旨

本論文では、文字方向および行方向に依存しないオンライン手書き文字列認識について述べる。

近年、タッチベースのスマートフォンやタブレット、電子ボード、そして、Anoto ペンや e-pen などのペン入力インタフェースの発展に伴い、直接指示・直接入力インタフェースが普通になってきている。さらに、これらの筆記面は大型化し、人々は大きい記入面に自由に書けるようになってきており、少ない筆記制限で自由に筆記された手書き文字列認識に関心が移行しつつある。このような記入平面の環境において、日本や中国の人々はよく水平、垂直、そして斜めに文字列を記入する。

文字ごとに課された記入枠のある単独文字認識だけではなく、自由に記入された記入枠のない手書き文字列認識が必要とされる。そして、大きい記入平面に記入される手書き文字列を高い精度で認識するための要求が高まってきている。

オンライン枠なし手書き文字列認識における既存の研究では、左から右への方向の横書き文字列を対象としたものがほとんどである。一方、日本や中国では、左から右への横書きと上から下への縦書きはよく利用され、混在も相当ある。また、斜め書きの文字列さえ出現している。研究としては、どのような文字方向でも、また、どのような行方向でも認識できる方法を確立しておく必要がある。

本研究では、ブロックグループ手法を使って文字列構成要素に分割し、各文字列構成要素に対して文字方向推定して正規化し、文字方向正規化された文字列を文字列方向量子化し、オンライン手書き文字列のストロークから多次元の特徴値を抽出し、そして、仮分割をして仮切り出しポイントを生成し、仮切り出しポイントをさらに多次元の特徴値にSVMの手法を適用することで、文字列の切り出しポイント候補を生成する。文字列分割候補が生成された後、正確な認識結果を得るためには、パス評価をする必要がある。パス評価には、文字認識エンジンから得た得点、文字列パタンのサイズ、位置関係、幾何学モデル、文脈などの情報を統合して、Viterbi 探索を行って、最適な文字列分割結果及び認識結果を判定する。

上記の手法をオンライン手書き文字パターンデータベース (*HANDS- Kondate_t_bf-2001_11*) に適用した結果、右向き文字列の認識率は92.22%であり、左向き文字列の認識率は92.93%、下向き文字列の認識率は91.60%、上向き文字列の認識率は91.52%になった。横書き、縦書き、斜め書きが混在する文字列を横書きに近い率で認識できるようになった。結果として、文字方向および行方向に依存しないオンライン手書き枠なし文字列認識について大幅な認識率向上を達成し、実用レベルになった。これは、タブレットPCや対話型

電子白板などでの自由な筆記認識に有効である。研究結果としてはアイラボ株式会社との共同研究である。

将来の課題として、最近の機械学習手法を取り入れてパターン処理・認識技術，畳み込みニューラルネットワーク(CNN)によるより良い精度の文字列認識を目指す。

第1章「緒論」では、本研究の研究背景と研究目的について述べる。そして、本論文の構成について述べる。

第2章「最新動向」では、本研究に関連する最新動向について述べる。つまり、従来のオンライン手書き文字列認識技術，オンライン文字方向および行方向自由の文字列認識技術，及び文字方向および行方向自由文字列認識の各研究分野について先行研究を紹介し，本研究の位置づけを明らかにする。

第3章「文字認識システム」では、デジタルペンなどから入力された時間情報を有するオンライン手書き文字パターンを処理する手書き文字認識システムについて述べる。ここでは，オンライン文字認識手法とオフライン文字認識手法を統合している。まず，多字種文字認識の高速化のために大分類についての概要を述べる。次に，オンライン手書き文字認識とオフライン手書き文字認識についてそれぞれの概要，及びそれを構築する各構成要素を述べる。そして，統合手法を述べる。

第4章「文字方向自由オンライン手書き文字列認識」では、文字方向の推定と仮定、行方向の推定、仮分割切出し手法に基づく文字方向自由オンライン手書き文字列認識手法について述べる。ここでは，文字列パターンに対するサポートベクトルマシン(SVM)を用いたストロークの分類方法，候補ラティスの生成方法，及び確率モデルによる文字列候補パスの評価基準とパラメーターの最適化について述べる。

第5章「言語の文脈処理と幾何学的な文脈処理」では、文字列認識精度を向上するために重要な評価要素技術である言語の文脈と幾何学的な文脈処理について述べる。

第6章「実験」では、評価実験の設定と実験結果，実験結果の分析について述べる。

第7章「結言」では、本論文の成果をまとめた上で，今後の課題について述べる。

Acknowledgements

First, I would like to express my gratitude to **Prof. Masaki Nakagawa** for giving me an opportunity to study in his laboratory. I was always led by their skillful guidance and helpful suggestions, which made it possible for me to go this far. The patience, dedication, and constant encouragement of my supervisors made it possible for me to deliver a dissertation of appreciable quality and standard. His responsible attitude is deeply appreciated and will guide my life in future.

Second, I would like to deeply thank **Dr. Bilan Zhu** for guiding me both in my study and in my life. She always gives me good advice when I have problem. She helped me to know well online recognizer for handwritten Japanese text.

Further, I would like to thank all of my teachers who assisted me with my study, and I would also like to thank all my friends, classmates and laboratory-mates, for their help and all the supports they provided. And I would like to express my gratitude to the common research company iLabo for supporting me in my study.

Finally, I am forever thanks to my family for their patience, understanding, encouraging me to concentrate on my study.

Content

目次

Abstract	2
論文要旨	4
Acknowledgements	6
Content	7
1. Introduction	10
1.1 Background	10
1.2 Objective	11
2. State of The Art	12
2.1 On-line Handwritten Text Recognition	12
2.1.1 Segmentation-based method	13
2.1.2 Integrated segmentation and recognition method	15
2.2 On-line Orientation Free Handwriting Text Recognition	23
3. Architecture of Japanese Text Recognizer	25
3.1 On-line Character Recognition	25
3.1.1 Introduction	25
3.1.2 Linear normalization	26
3.1.3 Feature points extraction	27
3.1.4 MRF for character recognition	28
3.2 Off-line Character Recognition	31
3.2.1 Introduction	31
3.2.2 Non-linear normalization	32

3.2.3	Directional feature extraction.....	34
3.2.4	Blurring and sampling	35
3.2.5	Dimensionality reduction	36
3.2.6	MQDF-based off-line character recognizer.....	37
3.3	Recognizer Combination.....	38
4.	Orientation Free On-line Handwritten Text Recognition System	40
4.1	Introduction	40
4.2	Line Direction and Character Orientation	40
4.3	Flow of Recognition Process	41
4.4	Segmentation of Handwriting into Text Line Elements.....	42
4.4.1	Line segmentation algorithm	43
4.5	Estimation and Normalization of Character Orientation.....	48
4.5.1	Estimation of character orientation	48
4.5.2	Assumption of character orientation	50
4.6	Estimation and Quantization of Line Direction.....	50
4.7	Over-Segmentation.....	51
4.7.1	Stage 1: Classification by two features.....	52
4.7.2	Dimensionality reduction (PCA)	54
4.7.3	SVM classification	55
4.8	Construction of SR-Lattice.....	60
4.9	Search and Recognition	61
4.10	Optimization of Parameters	64
4.10.1	MCE.....	64
4.10.2	GA	66

5. Linguistic Context and Geometric Context	67
5.1 Linguistic Context	67
5.1.1 N-gram language model	69
5.1.2 Smoothing algorithm	70
5.2 Geometric Context	72
6. Experiments	75
6.1 Kondate Database	75
6.2 Datasets	76
6.3 Evaluation of Dimensionality Reduction	80
6.4 Evaluation of Parameter Optimization by GA	81
6.5 Comparison with the Segmentation-updated System	82
6.6 Performance on Arbitrary Line Direction and Character Orientation	83
6.7 Performance on Artificially Rotated Text lines	85
6.8 Evaluation of Time Cost and Memory Requirement for Dictionaries	86
7. Conclusion and Future Work	88
7.1 Conclusion	88
7.2 Future work	88
Reference	91
Appendix- I : List of Figure	101
Appendix- II : List of Tables	103
Appendix-III: Author's Publications	104

1. Introduction

1.1 Background

Due to the wide spread of tablets, electronic whiteboards, and digital pens as well as the expansion of touch-based smart phones, users can write more freely as on a piece of paper and input handwriting into computers. Such freely handwritten text brings new challenges to remove writing constraint from on-line handwriting recognition. Asian people whose languages are Chinese origin often write text horizontally, vertically or even slantingly in a mixed way. Moreover, users may draw figures and write text along slanted lines in figures. Separation of text and non-text is treated in other papers [1, 2] and this paper focuses on the recognition of text of arbitrary line direction and character orientation.

Most of the previous publications and systems for on-line handwritten text recognition assume either horizontal or vertical lines of text [3-9], while we are trying to erase all writing constraint from on-line text input. We proposed a model to recognize mixtures of horizontal, vertical and slanting lines of text with arbitrary character orientation [10] and implemented a system [11]. Then, we improved segmentation by SVM for arbitrary line direction but normal character orientation [12]. We call it the segmentation-updated system. Unfortunately, its performance was inadequate for real use. Jin et al. proposed a line-direction free method for on-line unconstrained cursive handwritten Chinese word recognition while assuming normal character orientation [13]. It was designed for short handwritten text line recognition and the gravity center information of characters was used to detect the line direction. For both line-direction-free and character-orientation-free recognition, however, the gravity centers are not adequate to detect the both.

Unlike isolated character recognition, handwritten string recognition faces the difficulty of character segmentation because characters cannot be reliably segmented before they are recognized. Moreover, characters tend to be written more cursively. On top of the segmentation problem, handwritten text recognition of arbitrary line direction and character orientation must assure high recognition rate for any line direction and character orientation, and it must perform smoothly even where line direction and character orientation change. Moreover, its recognition speed must be quick enough on a usual platform and its memory requirement must be not too large compared from recognizers of horizontally handwritten text.

This paper we present an updated system for on-line handwritten Japanese text of arbitrary line direction and character orientation. We follow the over-segmentation-based approach due to

its advantages against the segmentation-free approach as discussed in [14]. The over-segmentation approach is to segment wherever characters must be segmented but may segment individual characters, which can be merged when they are recognized.

1.2 Objective

The research objective is to develop a handwriting recognizer based on the existing system [10, 11], which can recognize line direction free and character orientation free handwritten Japanese text.

The Japanese is a large character set language, which includes thousands of ideographic characters of Kanji, two sets of phonetic characters (Hiragana and Katakana), alphanumeric, and symbols. Most Kanji character patterns are composed of multiple sub patterns called radicals, which are shared among many Kanji character patterns.

Due to the text lines are free from line direction and character orientation, the unstable line direction and character orientation it is a challenging research work to segment text line correctly and segment characters correctly for every line direction before recognition as human for machine. To solve the problem, we attempt to:

- Text line segmentation

Segment text lines compose of horizontal, vertical and slanted lines of text with arbitrary character orientation into text line elements.

- Over-segmentation

Decide segmentation points and non-segmentation points in quantized 4 directions using the two-stage classification scheme.

- Evaluate the likelihood of candidate segmentation paths.
- Train and decide the weight of each factor automatically by a genetic algorithm.
- Optimal path can be found by the Viterbi search.

2. State Of The Art

In this chapter, we mainly review the state-of-the-art recognition methods for on-line handwritten text recognition, and on-line handwriting line direction free and character orientation free text recognition

2.1 On-line Handwritten Text Recognition

With the development of pen-based or touch-based devices, such as tablet PCs, digital pens and electric whiteboards and so on, the writing area of these devices becomes larger than before. People tend to write text continuously with little constraints. The demand for improving the handwriting text recognition is still increasing to meet potential many applications. On-line handwritten text recognition has been receiving larger attention, especially for unconstrained text recognition.

In general, handwritten text pattern recognition methods divided into on-line recognition and off-line recognition [15]. On-line recognition recognizes text patterns captured from a pen-based or touch-based input device where a series of trajectories of pen-tip or finger-tip movements are recorded, while off-line recognition recognizes text patterns captured from a scanner or a camera device as two dimensional images. Due to the on-line handwritten text pattern includes both temporal information of pen-tip or finger-tip movements and spatial shape information, the on-line handwriting recognition can yield higher recognition accuracy than off-line recognition. Moreover, on-line handwriting recognition provides friendly interaction and adaptation capability for users, such as the recognition result is showed and updated at the same time while writing, user can respond to the recognition result to correct misrecognition.

The research on on-line handwriting recognition started in the 1960s and has been receiving intensive interest from the 1980s. Tappert et al. [16] made a comprehensive survey before the 1990s. Nakagawa gave a survey focused on on-line handwritten Japanese characters recognition [17]. Since the 1990s, the research efforts have been aiming at the relaxation of constraints to ensure successful recognition, such as writing in boxes and the compliance with standard shapes. In recent survey papers, Plamondon et al. [15] mainly reviewed the advances of western handwriting recognition. Liu et al. [18] reviewed the advances in on-line Chinese and Japanese handwriting recognition from the 1990s. Recently, Zhu et al. [19] reviewed the on-line handwriting Japanese character recognition and its practical applications.

The handwritten Japanese/Chinese text recognition is more challenging than western

language due to the large character set. Japanese character set consists of various characters: symbols, numerals, hiragana and katakana (called Kana), and Kanji characters of Chinese origin. Hiragana and katakana are phonetic characters. Kanji characters are ideographic characters, which have divided into two classes: JIS (Japanese Industrial Standard) first level set and JIS second level. The JIS first level set contains 2,965 common use characters, which are necessary for reading the newspaper, and the JIS second level set contains 3,390 characters less common and special characters for naming.

Chinese characters sets consist of traditional Chinese characters mainly used in Taiwan, and simplified Chinese characters used in the mainland of China. The simplified Chinese characters includes two character sets, one contains 3,755 characters and the other contains 6,763 characters, where the first set is a subset of the second one, were announced as the National Standard GB2312-80. The traditional Chinese set includes 5,401 characters. In both simplified and traditional Chinese, about 5,000 characters are frequently used [18].

Moreover, most Kanji/Chinese character patterns are composed of multiple sub patterns, called radicals, which are shared among many Kanji character patterns. In Kanji character patterns, some are simple consisting of a single radical, while others are complex with multiple radicals.

In addition, the various writing styles also obstruct handwritten text recognition. The handwritten scripts are generally classified into three typical styles: regular style, fluent style and cursive style. The regular style is also referred to as block style or hand-printed style, which is written carefully with keeping fairly strict proper stroke number and order. The fluent style is often called “cursive” style, which is close to peoples’ practical writing and is written faster with fewer strokes, and some characters are connected together. The current recognition systems can recognize regular script with high accuracy, whereas the recognition of fluent or cursive style still remains unsolved and requires more intensive research efforts. The fluent or cursive script is the target of most recognition systems, which features greater variability of stroke-order and stroke-number within character and occurs frequently in practical writing.

Therefore, it is impossible to segment characters unambiguously in handwritten text recognition. Many works have focused on resolving the segmentation problem. These proposed methods can be roughly classified into the following categories: segmentation-based method, and integrated segmentation and recognition method.

2.1.1 Segmentation-based method

The segmentation-based method attempts to segment characters before character

recognition solely according to geometric layout features, such as character size, position, and inter-relationship.

Tseng et al. [20] proposed a segmentation method based on merging strokes and dynamic programming for the off-line handwritten Chinese characters recognition. It firstly extracts the strokes of the off-line characters to build the stroke bounding boxes. Then, the stroke bounding boxes are heuristically merged as a candidate character or a part of candidate character pattern using knowledge-based merging operations. Finally, the best segmentation boundaries are found by dynamic programming method. This method, however, is feasible only for neatly handwritten text. The segmentation performance mainly relies on the extracting stroke algorithm from characters.

Lu et al. [21] proposed a method to segment handwritten Chinese destination addresses of mail pieces. It merges subassemblies of Chinese characters based on the structural features of Chinese characters and the topological relations of subassemblies, namely, left-right, upper-lower and inside-outside relations. This pure structure-based segmentation method, however, is only suitable for handwritten text patterns without connected characters.

Zhao et al. [22] presented a two-stage approach to segment unconstrained off-line handwritten Chinese characters. In the first segmentation stage, according to the vertical projection and background skeleton, a horizontal handwritten Chinese character text is coarsely segmented into several blocks, and the blocks of connected characters are identified. The candidate segmentation points are found. In the second stage, connected characters are separated using geometric features of strokes, then the fine segmentation paths are extracted using fuzzy decision rules, which classify the candidate segmentation points. This segmentation method can resolve parts of connected characters. The segmentation accuracy of characters, however, is 81.6% on 1,000 unconstrained handwritten Chinese character texts. Wei et al. [23] proposed a new approach for connected Chinese characters, where the best segmentation path can be found by genetic algorithm.

Liang et al. [24] proposed a meta synthetic method to segment off-line handwritten Chinese character texts. For non-touching characters, it firstly applies the Viterbi algorithm to obtain the candidate segmentation paths, then a dynamic programming algorithm is applied to merge components. For touching characters, it firstly extracts candidate segmentation paths according to background and foreground information, and extracts peripheral features for each candidate segmentation path. Then the best segmentation path is found by the mixture probabilistic density function whose parameters are obtained by the EM algorithm.

Furukawa et al. [25] proposed a segmentation method for online unconstrained handwritten

Japanese texts using off-stroke (between strokes) features. In this method, the handwritten text is pre-segmented into basic segments, and a segmentation graph is constructed, where a node stands for a candidate segmentation point, an edge stands for a candidate character pattern, which is created by merging one or more basic segments. Then, it extracts the features of each candidate character pattern, which include temporal and geometric features, and proposed off-stroke features within candidate character patterns and between candidate character patterns. Based on the assumption that each feature distribution fits a normal distribution, the candidate segmentation pattern likelihood can be calculated from these extracted features using a probabilistic model. Finally, an optimal segmentation path on the segmentation graph is found by dynamic programming (DP). The character segmentation rates, however, is 75.6% of all characters.

2.1.2 Integrated segmentation and recognition method

Handwritten Japanese/Chinese text recognition is a challenging problem due to the fact that spaces between characters are not obvious, and many Kanji characters comprise radicals with internal gaps, as well as character touching. Without character recognition cues and linguistic context, characters in handwritten text patterns cannot be segmented unambiguously. A feasible solution to overcome the ambiguity of character segmentation is called the integrated segmentation and recognition method. Liu et al. [26] evaluated several common pattern classifiers based on this integrated segmentation and recognition framework, which includes neural classifiers, discriminative density models, and support vector classifiers, on handwritten numeral texts recognition. They demonstrate that superior text recognition performance can be achieved with appropriately designed classifiers even with simple pre-segmentation and without using geometric context in post-processing.

The integrated segmentation and recognition method is classified into segmentation-free and over-segmentation-based methods [27], [28]. The two methods are also called implicit segmentation and explicit segmentation methods, respectively. Segmentation-free methods will be introduced in the next section.

Over-segmentation-based methods [8], [10], [29], [30], [31], [32], [33], [34], attempt to split character patterns at their true boundaries and classify the split character patterns. Character patterns may also be split within them, but they are merged later. This is called over-segmentation. The over-segmentation-based method is mainly accomplished in two steps: over-segmentation and path search. The handwritten text pattern is firstly over-segmented into primitive segments, and each segment composes a single character or part of a character. The primitive segments are combined to generate candidate character patterns, and then a

segmentation lattice is constructed as shown in Figure 2-1, where a node stands for a candidate segmentation point and an edge stands for a candidate character pattern. Each candidate character pattern can obtain several similar character classes with the corresponding class scores by character recognition, and then segmentation-recognition candidate lattice is constructed, where each path in the lattice corresponds to segmentation-recognition paths (hypothesis), which is evaluated by combining the character recognition, linguistic context and geometric context. Finally, the optimal recognition result text is found by searching for the optimal segmentation-recognition path with maximum score or minimum cost.

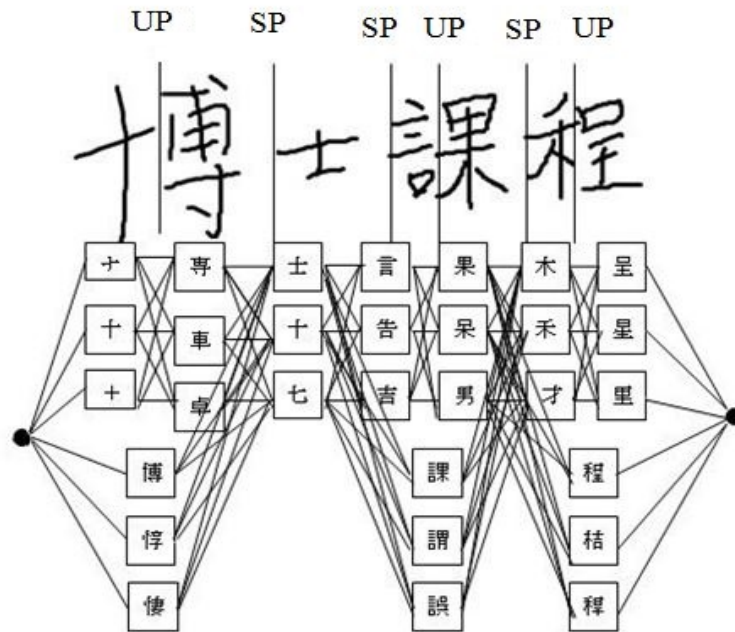


Figure2-1 Segmentation lattice. (SP is segmentation point and UP is undecided point.)

(1) Path evaluation

The key issue in over-segmentation-based text recognition is how to evaluate of candidate segmentation-recognition paths (segmentation hypotheses) in the candidate lattice. A desirable criterion should make the path of correct segmentation have the maximum score. Probabilistic model based on the maximum a posteriori (MAP) criterion [35] is one of the frequently used methods for segmentation hypothesis evaluation [8], [36], [37].

An early text class probability model can be found in [38]. Assume that a handwritten text pattern X is segmented into a sequence of segments $\mathcal{S} = s_1 s_2 \dots s_n$ (note that there are many segmentation candidates even with the same text length), where s_i stands for a candidate character pattern, and is assigned to a text class $\mathcal{C} = c_1 c_2 \dots c_n$, where character c_i is assigned to

s ; by a character recognition. The a posteriori probability of the text class is defined as:

$$P(C|X) = \sum_S p(C, S|X) \quad (2-1)$$

The segmentation candidate is constrained to have the same length of C , that is $|S|=|C|=n$. The candidate character patterns are represented by the feature vectors $X=x_1x_2, \dots, x_n$. To avoid summing over multiple segmentation candidates in Eq. (2-1), the optimal text class can be decided by

$$C^* = \arg \max_c \max_s P(C, S|X)$$

This is to find for the optimal segmentation candidate S for each text class. Using the Bayesian law, $P(C, S|X)$ is decomposed into

$$P(C, S|X) = \frac{P(X|C, S)P(C, S)}{P(X)} = \frac{P(X|C, S)P(S|C)P(C)}{P(X)} \quad (2-3)$$

Assuming context independence of character shapes, it can be approximated as:

$$\begin{aligned} P(C, S|X) &\approx P(C) \prod_{i=1}^n \frac{P(x_i|c_i, s_i)P(s_i|c_i)}{P(x_i)} \\ &= P(C) \prod_{i=1}^n \frac{P(c_i s_i | x_i)}{P(c_i)} \\ &= P(C) \prod_{i=1}^n \frac{p(c_i | s_i, x_i) P(s_i | x_i)}{P(c_i)} \end{aligned} \quad (2-4)$$

where $P(s_i|x_i)$ stands for the probability of geometric context, the priori probability of text class

$P(C)$ stands for the linguistic context. It is often approximated by a bigram language model for an open vocabulary:

$$P(C) = P(c_1) \prod_{i=2}^n P(c_i|c_{i-1}) \quad (2-5)$$

Assume that the character recognition is not related into the geometric context, $P(c_i|s_i, x_i)$ can be replaced by $P(c_i|x_i)$. Ignoring the geometric context, $P(s_i|x_i)$ can be viewed as a constant, and the text class probability in Eq. (2-4) is approximately

$$P(C, S|X) \approx P(C) \prod_{i=1}^n \frac{P(c_i|x_i)}{P(c_i)} \quad (2-6)$$

where $P(c_i|x_i)$ stands for the posterior probability of the candidate character pattern x_i being recognized as c_i . In literature [38], $P(c_i|x_i)$ is approximated by the output of a multi-player percept on (MLP) classifier.

In handwritten Japanese text recognition, Nakagawa et al. [10] proposed a text class probability model incorporating the geometry of inter-character gap. The candidate pattern sequence is denoted by $S = s_1 g_1 s_2 g_2 \dots s_n g_n$, where s_i represents the geometric features of the i -th character pattern, which includes the width and height of bounding box, and g_i represents the geometric features between adjacent two character patterns. In the Eq. (2-3), $P(X)$ is omitted because it is independent of text class. $P(C)$ is estimated by a bigram model. Hence, $P(C, S|X)$ is approximated by

$$\begin{aligned} P(C, S|X) &= P(X|C, S)P(S|C)P(C) \\ &\approx \prod_{i=1}^n P(x_i|c_i) \times \prod_{i=1}^n P(s_i|c_i) P(g_i|c_i c_{i+1}) \times \prod_{i=1}^n P(c_i|c_{i-1}) \end{aligned} \quad (2-7)$$

where $P(x_i|c_i)$ is the likelihood of pattern x_i with respect to class c_i , which is estimated by a character classifier. $P(s_i|c_i)$ and $P(g_i|c_i c_{i+1})$ can be seen as character likeliness and

between-character compatibility, respectively. Finally, by taking log of the both sides in Eq. (2-7), the all score of a path is the summation of product of probabilistic likelihood in the right-hand side. The literature [31], [39], [40], [41] have used the similar evaluation criterion.

If the character classifier is trained to be resistant to non-characters, namely, all defined classes are assigned low confidence values on non-character patterns. Without geometric context score, it can still give high text recognition accuracy [18]. The text pattern is classified to

$$C^* = \arg \max_c P(C)P(X|C) = \arg \max_c P(C)P(X|C) \prod_{i=1}^n P(x_i|c_i) \quad (2-8)$$

By assuming $P(C)$ is equal, the classification criterion is further simplified to

$$C^* = \arg \max_c P(X|C) = \arg \max_c \prod_{i=1}^n P(x_i|c_i) \quad (2-9)$$

A text pattern can be segmented into variable lengths of character pattern sequences. However, since the likelihood measure is usually smaller than one, the summation criterion is often biased to paths with fewer characters, namely short path. This will raise the segmentation error of merging multiple characters into one candidate pattern. To overcome this bias, Tulyakov et al. [42] proposed a normalized text probability score as follows:

$$C^* = \arg \max_c \left(\prod_{i=1}^n P(x_i|c_i) \right)^{1/n} \quad (2-10)$$

The normalized criterion, obtained by dividing the summation criterion by the number of segmented characters (segmentation length), tends to over-split characters.

To solve the problems, Zhu et al. [8] proposed a robust context integration model for on-line handwritten Japanese text recognition. By labeling primitive segments, the proposed path evaluation criterion can not only integrate the character shape information into recognition by introducing some adjustable parameters, but also is insensitive to the number of segmented character patterns because the summation is over the primitive segments. The path evaluation

criterion is expressed as follows:

$$\begin{aligned}
f(X, C) = & \sum_{i=1}^n \left\{ \sum_{h=1}^6 [\lambda_{h1} + \lambda_{h2}(k_i - 1)] \log P_h + \lambda_{71} \log P(g_{ji}|SP) \right. \\
& \left. + \lambda_{72} \sum_{j=j_i+1}^{j_i+k_i-1} \log P(g_j|NSP) \right\} + m\lambda
\end{aligned} \tag{2-11}$$

where $P_1, P_2, P_3, P_4, P_5,$ and P_6 stands for the probabilities of trigram ($P(c_i|c_{i-2}c_{i-1})$), character pattern sizes ($P(b_i|c_i)$), inner gaps ($P(q_i|c_i)$), single-character positions ($P(p_i^u|c_i)$), pair-character positions ($P(p_i^b|c_{i-1}c_i)$) and character recognition ($P(x_i|c_i)$), respectively. k_i is the number of primitive segments contained in the candidate character pattern x_i . $\lambda_{h1}, \lambda_{h2}$ ($h=1\sim 7$) and λ are the weighting parameters. g_i is the between-segment gap feature vector. If the adjacent two segments is within a true character, the label is NSP (non-segmentation point), otherwise is SP (segmentation point). Due to the character recognition is estimated by the combination score of on-line and off-line isolated character recognizers, Zhu et al. [43] divided the character recognition into two parts $P(x_{on}^i|c_i)$ and $P(x_{off}^i|c_i)$, where x_{on} denotes the on-line features of x_i , x_{off} denotes the off-line features of x_i . $P(x_{on}^i|c_i)$ and $P(x_{off}^i|c_i)$ are estimated by the score of the on-line recognizer and off-line recognizer, respectively. Then the path evaluation criterion in Eq. (2-11) is changed as follows:

$$\begin{aligned}
f(X, C) = & \sum_{i=1}^n \left\{ \sum_{h=1}^7 [\lambda_{h1} + \lambda_{h2}(k_i - 1)] \log P_h + \lambda_{81} \log P(g_{ji}|SP) \right. \\
& \left. + \lambda_{82} \sum_{j=j_i+1}^{j_i+k_i-1} \log P(g_j|NSP) \right\} + m\lambda
\end{aligned} \tag{2-12}$$

Under this same path evaluation criterion, Gao et al. [44], [45] reduced the text recognizer size for hand-held devices by compressing each component in this text recognition system. It compresses MQDF2 based off-line character recognizer by linear discriminant analysis (LDA), vector quantization and data type transformation, and selects an elastic matching based on-line recognizer. This recognition method has been successfully applied in smart phones and tablets.

In handwritten Chinese text recognition, to overcome the problem of sensitivity of the path length, Wang et al. [46] used the similar path evaluation criterion for real-time recognition of on-line handwritten sentences. The path evaluation is the combination of multiple contexts as follows:

$$\begin{aligned}
f(X, C) = & \sum_{i=1}^n \{k_i \log P(c_i|x_i) + \lambda_1 \log P(c_i|c_{i-1}) + \lambda_2 \log P(c_i|g_i^{uc}) \\
& + \lambda_3 \log P(z_i^p = 1|g_i^{ui}) + \lambda_4 \log P((c_{i-1}, c_i|g_i^{bc}) \\
& + \lambda_5 \log P(z_i^g = 1|g_i^{bi})\}
\end{aligned} \tag{2-13}$$

where $P(c_i|x_i)$ is given by the character classifier, $P(c_i|c_{i-1})$ is a bigram language model, $P(c_i|g_i^{uc})$ and $P(z_i^g = 1|g_i^{ui})$ stand for the unary class-dependent (*uc*) and unary class-independent (*ui*) geometric score, respectively. $P(c_{i-1}, c_i|g_i^{bc})$ and $P(z_i^g = 1|g_i^{bi})$ stand for binary class-dependent (*bc*) and binary class-independent geometric score (*bi*), respectively. Compared to literature [33], they added unary and binary class-independent geometric information to evaluate the path.

Wang et al. [32] also used the similar path evaluation criterion [46] for off-line unconstrained handwritten Chinese text recognition. Paths are evaluated from the Bayesian decision view by combing character recognition scores, class-dependent and class-independent geometric contexts, and linguistic context. The recognition performance on the HIW-MW test set [47] achieved the character-level accurate rate of 91.86% and correct rate of 92.72% using word class bigram.

Li et al. [34] proposed a new probabilistic model for off-line unconstrained handwritten text recognition to evaluate possible segmentation hypotheses. The path evaluation criterion as shown in Eq. (2-13) can be implemented in a simply way that follows Bayesian rules using just two classifiers, one is MQDF based isolated character recognizer, which has been trained by a linear discriminant analysis (LDA) –based negative training strategy using non-character patterns, the other is a the character verifier to check whether a candidate character pattern is true character or not, which can be transformed to posterior probability of a five-class MQDF classifier, including Chinese class, digit class, punctuation class and two classes of non-characters. The proposed method achieved the character-level recognition rates of 80.15% with a bigram language model on HIT-MW test set.

Zhou et al. [9] proposed a new method for on-line handwritten Chinese/Japanese text recognition by defining the high-order semi-Markov conditional random fields (CRF) on the candidate lattice to directly estimate the posterior probability of segmentation-recognition paths. In this semi-CRF model, it fuses the scores of character recognition, geometric and linguistic contexts in a principled MAP framework. This method has yielded superior text recognition performance compared to the state-of-the-art methods on the test sets of CASIA-OLHWDB (Chinese) [48], TUAT Kondate (Japanese) and ICDAR 2011 Chinese handwriting recognition competition.

The weighting parameters in the path evaluation criterion were sometimes determined by trial and error to yield higher text recognition performance. In recent years, some works have applied the supervised text-level learning approach to estimate the weighting parameters by minimizing the text recognition error. Zhu et al. [8] optimized the weighting parameters for on-line handwritten Japanese text recognition using genetic algorithm (GA). They also compared with the minimum classification error (MCE) criterion [49] optimized by stochastic gradient decent [50], and showed that GA-based optimization method yields better text recognition performance than MCE. Wang et al. [46] optimized the combining weights by MCE learning for on-line handwritten Chinese text recognition. The parameters in MCE learning are learned by stochastic gradient decent. Zhou et al. [36] proposed learning the weights by minimizing the negative log-likelihood (NLL) loss under the framework of CRF, and compared its performance with MCE criterion. Zhou et al. [9] modified NLL loss by adding a margin term to improve the generalization performance of parameter learning in semi-CRF.

(2) Path search

The search of optimal path for handwritten Japanese/Chinese text recognition is not trivial due to the large number of candidate segmentation-recognition paths in the candidate lattice. Moreover, the search is complicated when using word-level language models because the word segmentation is again a combinatorial problem [32]. The exhaustive search strategy that computes the scores of all segmentation-recognition paths and then selects the optimal one is computationally expensive.

Heuristic search algorithms that evaluate only a portion of segmentation-recognition paths have been commonly used in handwritten text recognition. The speech recognition field has contributed many efficient search algorithms based on dynamic programming (DP) and beam search [51].

If the segmentation-recognition path is scored by the accumulated cost form, the optimal

path can be easily found by dynamic programming algorithm [8], [10], [39], [40], [41]. Under the normalized criterion, however, DP algorithm does not guarantee finding the optimal path. Beam search strategy has been employed. Among the partial paths ending at an intermediate node in the candidate lattice, beam search retains multiple partial paths with high scores for extension, the retained partial paths are also called beam width. All the retained partial paths of the parent nodes are extended to each child, where several high-score partial paths are again retained. At the terminal node, the path of highest score in the retained paths is as the optimal path. Liu et al. [26] used the beam search to find the optimal result for handwritten numeral text recognition.

On the other hand, according to the order of node generation in the heuristic research, the search algorithms can be divided into character-synchronous and frame-synchronous search [27]. The frame-synchronous is also called time-synchronous search. Liu et al. [52] proposed lexicon-driven text recognition approach for Japanese mail address reading using character-synchronous beam search strategy. The all address phrases are stored in a tri structure lexicon. Due to the beam search is used to expand all the nodes of same depth in the search space synchronously and proceeds by depth until there is no open node to expand, the character-synchronous beam search is appropriate for lexicon-driven text recognition. Zhu et al. [53] proposed lexicon-driven approach for on-line handwritten Japanese disease names recognition using frame-synchronous beam search. It restricts the character categories of recognizing each candidate character pattern from the tri lexicon of disease names and preceding paths during path search, as well as the length of disease names. The beam search is used to expand all the nodes of same segment in the search space.

2.2 On-line Orientation Free Handwriting Text Recognition

Onuma et al. [11] proposed an on-line handwritten Japanese text recognition system that is liberated not only from writing boxes or rules lines but also from constraints on line direction and character orientation. This system first separates freely written text into text line elements, second estimates the line direction and character orientation, third hypothetically segment it into characters, fourth apply character recognition and finally select the most plausible interpretation by evaluating the likelihood, the method is working for a mixture of vertical, horizontal and skewed lines with arbitrary character orientations. But the recognition rate is not enough for the real use.

Jin et al. [13] proposed a method for on-line unconstrained cursive handwritten Chinese word recognition. By a novel gravity center balancing method, the orientation ranging from 0°

to 360° of handwritten words can be detected. After stroke extraction and breaking the strokes which may belong to two characters, over-segmentation is performed by a heuristic merging of strokes. By searching the paths generated from the over-segmentation result, considering both recognition and lexicon information, the handwritten word with characters even connected or partially overlapped can be recognized.

Jin et al. [13] proposed a line-direction free method for on-line unconstrained cursive handwritten Chinese word recognition while assuming normal character orientation [13]. It was designed for short handwritten text line recognition and the gravity center information of characters was used to detect the line direction. For both line-direction-free and character-orientation free recognition, however, the gravity centers are not good enough to detect the both.

Chiang et al. [54] present a general text recognition technique to handle non-homogeneous text by exploiting dynamic character grouping criteria based on the character sizes and maximum desired string curvature. This technique can be easily integrated with classic OCR approaches to recognize non-homogeneous text. In our experiments, we compared our approach to a commercial OCR product using a variety of raster maps that contain multi-oriented, curved and straight text labels of multi-sized characters. Their evaluation showed that their approach produced accurate text recognition results and outperformed the commercial product at both the word and character level accuracy. Text recognition is difficult from documents that contain multi-oriented, curved text lines of various character sizes. This is because layout analysis techniques, which most optical character recognition (OCR) approaches rely on, do not work well on unstructured documents with non-homogeneous text. Previous work on recognizing non-homogeneous text typically handles specific cases, such as horizontal and/or straight text lines and single-sized characters.

3. Architecture of Japanese Text Recognizer

3.1 On-line Character Recognition

In this chapter, we describe the on-line character recognition, composed of pre-processing, feature points extraction and two types of on-line recognizer LTM and MRF. To guide the selection of on-line recognizer in compression of handwritten text recognizer, we further evaluate both on-line recognizer in recognition rate, memory cost and time cost.

3.1.1 Introduction

The study of on-line character recognition started in the 1960s and has been receiving intensive concerns from the 1980s. Early works of on-line Japanese character recognition have been reviewed in [17] [18]. The newest survey described in [55]. The comprehensive survey of applying handwritten character recognition to PDA and Tablet begun before 1990 [56].

On-line character recognition mainly focus recognizing pen-based and touch-based input handwritten character patterns, often with writing box limitation for character pattern input. The on-line pattern is sequence of coordinates of pen-tip which sample the coordinates along the path from pen-down to pen-up.

The on-line handwriting recognition has a number of distinguishing features, which can be exploited to get more accurate results:

1. It is a real time process. It captures the temporal and dynamic information of the pen trajectory. This information consists of the number and order of pen-strokes, the direction of the writing for each pen-stroke and the speed of the writing within each pen stroke.

2. Very little pre-processing is required. The operations, such as smoothing, de-slanting, de-skewing, detection of line orientations, corners, loop and cusps are easier and faster with the pen trajectory data than on pixel images.

Since the each coordinate point of character patterns is extracted in real-time, it includes not only the location information and time cost of each stroke but also the order of strokes. Although more information of character pattern stored in on-line pattern, the off-line character recognition obtain the better recognition results in the character recognition literature. Nevertheless, now, Liu etc. have proposed novel method of directly extracting off-line feature vector from an on-line pattern, which ensure the off-line recognition method can be employed to enhance the on-line character recognition, together with on-line character recognizer. Therefore,

in this paper, we use on-line and off-line combined character to character recognition for text recognition. The off-line recognition methods will be described in Chapters 3.

3.1.2 Linear normalization

Linear normalization is considered to be the most important pre-processing factor for on-line character recognition. In fact, linear normalization is linearly mapped the character pattern onto a standard plane by interpolation or extrapolation. The size and position of character is controlled such that normalized plane in x and y dimension is filled. The implementation of interpolation/extrapolation is influential to the recognition performance [56, 57]. After linear mapping, the character pattern is not deformed except the aspect ratio changes.

For ease of feature extraction and classification, it is better to fill both dimensions of normalized pattern (standard pattern). However, in this case, the deformation is enlarged. In aspect ratio adaptive normalization (ARAN), however, the dimensions of the standard plane are not necessarily filled [58]. Depending on the aspect ratio, the normalized image is centered in the plane with one dimension filled. Assume the standard plane is square and the side length is denoted by L . Denote the width and height of the normalized character image as W_2 and H_2 , the aspect ratio is defined by

$$R_2 = \begin{cases} W_2/H_2, & \text{if } W_2 < H_2 \\ H_2/W_2, & \text{otherwise} \end{cases} \quad (3-1)$$

The normalized pattern is filled one dimension by $\max(W_2, H_2) = L$. That is, to keep the aspect ratio unchanged, the normalized image does not necessarily fill both dimensions. According to direction of mapping, the linear normalization can be divided into the forward mapping and backward mapping. Table 3-1 lists the coordinates mapping of linear normalization. a and β are computed by Eq. (3-2).

Table3- 1 Mapping of linear normalization.

Method	Forward mapping	Backward mapping
Linear	$x' = ax$	$x = x'/a$
	$y' = \beta y$	$y' = y/\beta$

$$\begin{cases} a = W_2/W_1 \\ \beta = H_2/H_1 \end{cases} \quad (3-2)$$

3.1.3 Feature points extraction

To reduce the computation complexity and discard repeated sampled coordinates, for on-line character recognition, using the feature points to express the on-line pattern rather than original coordinate sequence of the pattern is proved effective and efficient. Before feature extraction, the input pattern is normalized to 128×128 pixels by linear normalization described as section 3.1.2. For each stroke, first, the start and end points are picked up as feature points; then, the point farthest from the straight line through adjacent feature points is selected as a feature point while the distance is greater than a threshold. This process continues recursively until no more feature points are selected [59]. The process of feature extraction is shown by Fig. 3-1.

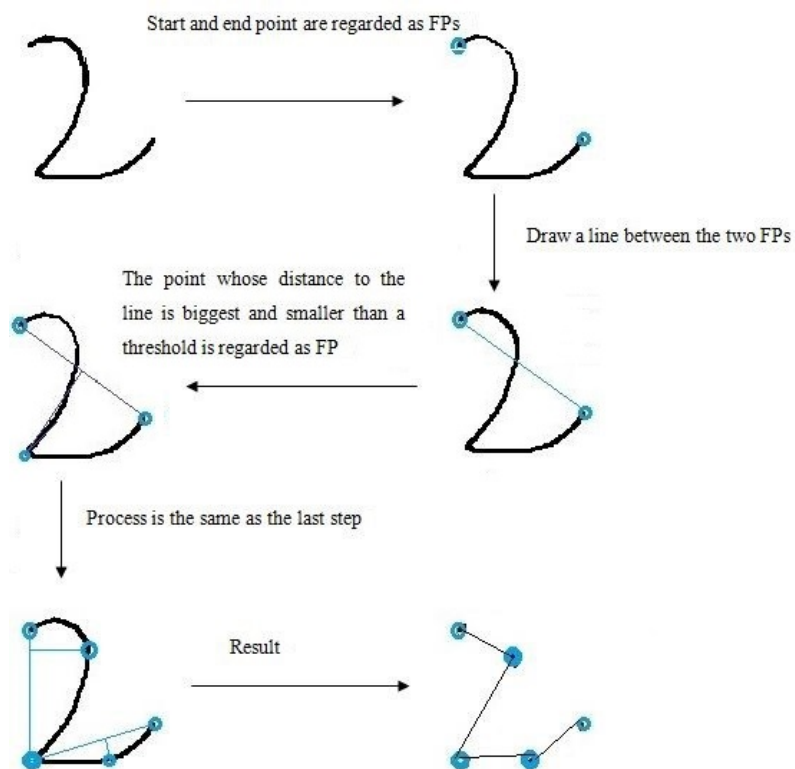


Figure3- 1 Process of feature points extraction

3.1.4 MRF for character recognition

MRFs can effectively integrate the information between neighboring pen-points such as binary features and triple features [60] and they have been successfully applied to Off-line handwritten character recognition [61] and on-line stroke classification [62]. However, MRFs have not been applied to on-line handwritten character recognition; current on-line handwritten character recognition tends to use HMM-based models (note that HMMs can be viewed as a specific case of MRFs).

Cho et al. [63] propose a Bayesian network (BN) based framework for on-line handwriting recognition. BNs share similarities with MRFs. BNs are directional acyclic graphs and model the relationships between the neighboring pen points as conditional probability distributions, while MRFs are undirected graphs and model the relationships between the neighboring pen-points as probability distributions of binary or triple features. Introducing weighting parameters to MRFs and optimizing them based on CRFs [64] or MCE [65] may bring even higher recognition accuracy; CRF has been successfully applied to on-line string and off-line word recognition [66, 67].

(1) Overall process of character recognition by MRF

The process of using MRF to on-line character recognition is described as follows. The input pattern is linearly normalized to standard size with aspect ratio unchanged. Then, we extract feature points by the algorithm proposed by Rammer, described in Section 3.3. Using the extracted feature points to express the structure of an input pattern is effective and efficient than using all pen-tip coordinate points sampled along the pen moving trace. Since in such case, it can not only improve the accuracy but also low the time cost of recognition [68] [69]. At last, we employ MRF model to match the feature points of input unknown character pattern with the states of each character class and obtain a similarity between input character pattern and each character class. Finally, we consider the character class with the largest similarity as the recognition result.

(2) MRF for character recognition

We note feature points from an input pattern as sites $S = \{s_1, s_2, \dots, s_l\}$ and states of a character class C as labels $L = \{l_1, l_2, \dots, l_j\}$. The mapping from S to L during character recognition notes as $F = \{s_1 = l_i, s_2 = l_j \dots, s_l = l_k\}$ called as a configuration.

The feature vector extracted from feature points of an input pattern is considered as the observation set \mathbf{O} . According to Bayesian theorem, the recognized character class is given by

Eq. (3-3).

$$C^* = \arg \max_C P(C|\mathbf{O}) = \arg \max_C P(C)P(\mathbf{O}|C) \quad (3-3)$$

Since exist more than one configuration F , $P(\mathbf{O}|C)$ can be further given by Eq. (3-4).

$$P(\mathbf{O}|C) = \sum_{\text{all } F} P(\mathbf{O}, F|C) \quad (3-4)$$

Making the matching under all F s is intractable, so we just consider the best figuration obtained by Viterbi algorithm. That is

$$P(\mathbf{O}|C) = P(\mathbf{O}, F_{\text{best}}|C) = P(F|C)P(\mathbf{O}|F_{\text{best}}, C) \quad (3-5)$$

The Hammersley-Clifford theorem establishes the equivalence between the Markov random fields [28].

$$P(F|C) = \frac{1}{Z} \exp(-E(F|C)) \quad (3-6)$$

where $E(F|C) = \sum_{cl} V_{cl}^F(F|C)$ is called the prior energy function and $V_{cl}^F(F|C)$ is called prior clique potential function defined on the corresponding cl . $Z = \sum_F \exp(-E(F|C))$ is the normalization factor called partition function.

Taking $P(\mathbf{O}, F|C)$ into consideration, we can obtain the global likelihood energy function given by Eq. (3-7). $E(\mathbf{O}|F_{\text{best}}, C)$ is computed by Eq. (3-8) where $V_{cl}^O(\mathbf{O}|F, C)$ is called the likelihood clique potential function.

$$P(F|C)P(\mathbf{O}|F_{\text{best}}, C) = \frac{1}{Z} \exp(-E(F|C) - E(\mathbf{O}|F_{\text{best}}, C)) \quad (3-7)$$

$$E(\mathbf{O}|F, C) = \sum_{cl} V_{cl}^O(\mathbf{O}|F, C) \quad (3-8)$$

For simplicity, we consider only single-site cliques $cl_1 = \{s_i\}$ and pair-site cliques $cl_1 = \{s_i, s_j\}$ to construct linear chain MRF. From above two equations, we can obtain Eq. (3-9), where l_{s_i} is the label of a class C assigned to s_i . O_{s_i} is the unary feature vector composed of x and y coordinates of site s_i , $O_{s_i s_j}$ is the binary feature vector composed of differences of x and y between sites s_i and s_j , extracted from the combination of s_i and s_j .

$$\begin{aligned}
E(F|C) + E(O|F_{\text{best}}, C) &= \sum_{cl} \left[V_{cl}^O(O|F, C) + \sum_{cl} V_{cl}^F(F|C) \right] \\
&= \sum_{s_i \in cl_1} [V_{cl_1}^O(O_{s_i}|l_{s_i}, C) + V_{cl_1}^F(l_{s_i}|C)] \\
&\quad + \sum_{\{s_i, s_j\} \in cl_2} [V_{cl_2}^O(O_{s_i s_j}|l_{s_i} l_j, C) + V_{cl_2}^F(l_{s_i} l_j|C)]
\end{aligned} \tag{3-9}$$

To derive the likelihood clique potentials from the negative logarithm of the conditional probabilities, we get the Eq. (3-10) from Eq. (3-9).

$$V_{cl_1}^O(O_{s_i}|l_{s_i}, C) = -\log P(O_{s_i}|l_{s_i}, C) \tag{3-10}$$

$$V_{cl_2}^O(O_{s_i s_j}|l_{s_i} l_j, C) = -\log P(O_{s_i s_j}|l_{s_i} l_j, C) \tag{3-11}$$

Moreover, since a label just interacts with only the neighboring labels, the state transition probability can be employed to evaluate the prior energy function instead of the prior clique potential.

$$E(F|C) = \sum_{i=1}^I -\log(l_{s_i}|l_{s_{i-1}}, C) \tag{3-12}$$

Therefore, the energy function is as follows:

$$\begin{aligned}
E(O, F|C) &= E(F|C) + E(O|F_{best}, C) \\
&= \sum_{i=1}^I \left[-\log P(O_{s_i}|l_{s_i}, C) - \log P(O_{s_i s_j}|l_{s_i} l_j, C) - \log P(l_{s_i}|l_{s_{i-1}}, C) \right] \quad (3-13)
\end{aligned}$$

$P(O_{s_i}|l_{s_i}, C)$ and $P(O_{s_i s_j}|l_{s_i} l_j, C)$ are evaluated by Gaussian functions. $P(l_{s_i}|l_{s_{i-1}}, C)$ is calculated as follows.

$$P(l_{s_i}|l_{s_{i-1}}, C) = \frac{\text{Number of transitions from } l_{s_{i-1}} \text{ to } l_{s_i}}{\text{Number of sites assigned } l_{s_{i-1}}} \quad (3-14)$$

$$P(l_{s_1}|l_{s_0}, C) = \frac{\text{Number of } s_1 \text{ to } l_{s_i}}{\text{Number of } s_1} \quad (3-15)$$

To train the MRF of each character class, we first initialize the feature points of an arbitrary character pattern among the training patterns of the character class as states of the MRF, set each unary feature vector of each feature point as the mean of the Gaussian function for each single-state, and set each binary feature vector between two adjacent feature points as the mean of the Gaussian function for each pair-state, and initialize the variances of those Gaussian functions and the state transition probabilities with 1. Then we use the Viterbi algorithm or the Baum-Welch algorithm to train the parameters of the MRF (the means and variances of Gaussian functions and the state transition probabilities). We repeat the training until the optimal parameters are obtained.

3.2 Off-line Character Recognition

3.2.1 Introduction

Off-line character recognition is known as Optical Character Recognition (OCR), because the image of handwriting pattern is converted into bit map pattern by an optically digitizing device such as optical scanner or camera. The recognition is done on this bit map pattern data for machine-printed or hand-written text. The research and development is well progressed for the recognition of the machine-printed documents. In recent years, the focus of attention is

shifted towards the recognition of hand-written script.

The major advantage of the off-line recognizers is to allow applying the electronic image process technology, including non-linear normalization and off-line feature. In recent years, nonlinear normalization (NLN) based on line density equalization, moment normalization (MN), bi-moment normalization (BMN), modified centroid-boundary alignment (MCBA), and their pseudo-two-dimensional (pseudo 2D) extensions all obtained good accuracy in handwritten character recognition. Moreover, as to off-line feature, the directional density feature and gradient feature extracted from character pattern also show more robust than feature points extracted directly from on-line pattern.

3.2.2 Non-linear normalization

Normalization regulates the size, position, and shape of character pattern, to reduce the shape variation between character patterns of the same class. Some strategies were proposed to deform the character shape with aim to reduce the within-class variation. The perspective transformation attempts to correct the imbalance of character width [70], the moment normalization attempts to rectify the rotation or slant [71], and the nonlinear normalization aims to equalize the line density [72, 73]. For slant normalization, the slant can also be estimated from character field context instead of moments [74]. In this thesis, we mainly focus on pseudo 2D bi-moment normalization

For the normalization of patterns, we employ the pseudo 2D normalization method. The coordinate mapping functions $x'(x, y)$ and $y'(x, y)$ are obtained by linearly combining one-dimensional functions with the weight depending on another dimension as given by Eq. (3-16). The one-dimensional functions are obtained by applying 1D normalization to the projection functions of partial images (for on-line pattern, it can be considered as a imaginary image on a plane).

$$\begin{cases} x'(x, y) = \sum_i w^i(y)x^i(x) \\ y'(x, y) = \sum_i w^i(y)x^i(x)' \end{cases} \quad (3-16)$$

For an on-line pattern which is considered as imaginary image $f(x, y)$ is partitioned into three horizontal soft strips by the weight function in y-axis:

$$f_x^i(x, y) = w^i(y)f(x, y), i = 1, 2, 3. \quad (3-17)$$

where $w^i(y)$ are weight functions as given by Eq. (3-18) and H_1 and y_c are boundary and coordinate in y-axis of centroid for the character pattern. W_0 is constant. Similarly, we obtain the three vertical soft strips. $f_y^i(x, y) = w^i(x)f(x, y), i = 1, 2, 3$.

$$\begin{cases} w^1(y) = w_0 \frac{y_c - y}{y_c}, y < y_c \\ w^2(y) = 1 - w^1(y), y \geq y_c. \\ w^3(y) = w_0 \frac{y_c - y}{H_1 - y_c}, y \geq y_c \end{cases} \quad (3-18)$$

The three horizontal strips $f_x^i(x, y), i = 1, 2, 3$ project onto the x-axis as in Eq. (3-19).

$$P_x^i(x) = \sum_y f_x^i(x, y), i = 1, 2, 3. \quad (3-19)$$

The projection functions of the three strips on x-axis, $P_x^i(x), i = 1, 2, 3$, are used to compute three coordinate functions $x^{(i)}(x)$, by using the bi-moment normalization (BMN). The three 1-dimensional coordinate functions are then combined into a 2D coordinate function as given by Eq. (3-20). The normalization composed of above two steps is commonly called P2DBMN.

$$x'(x, y) = \begin{cases} w^1(y)x^{(1)}(x) + w^2(y)x^{(2)}(x), y < y_c \\ w^3(y)x^{(3)}(x) + w^2(y)x^{(2)}(x), y \geq y_c \end{cases} \quad (3-20)$$

To obtain $x^{(i)}(x), i = 1, 2, 3$, by BMN, the second-order moments are split into two parts at the centroid: u_{20}^{i-} and u_{20}^{i+} in x-axis, u_{02}^{i-} and u_{02}^{i+} in y-axis. The bi-moments are computed from the projection of each strip as given by Eq. (3-21). The centroid of each strip is computed by Eq. (3-22). The boundaries of the input pattern are re-set to $\left[x_c^i - 2\sqrt{u_{20}^{i-}}, x_c^i + 2\sqrt{u_{20}^{i+}} \right]$ and $\left[y_c^i - 2\sqrt{u_{02}^{i-}}, y_c^i + 2\sqrt{u_{02}^{i+}} \right]$. For the x-axis, a quadratic function $u(x) = ax^2 + bx + c$ lings three points $\left(x_c^i - 2\sqrt{u_{20}^{i-}}, x_c^i, x_c^i + 2\sqrt{u_{20}^{i+}} \right)$ to normalized coordinate (0, 0.5, 1), and similarly, a quadratic function $v(y)$ works for the y-axis. Finally, the coordinate functions are given by Eq. (3-23).

$$\left\{ \begin{array}{l} u_{20}^{i+} = \frac{\sum_{x>x_c^i} (x - x_c^i)^2 P_x^i(x)}{\sum_{x>x_c^i} P_x^i(x)} \\ u_{20}^{i-} = \frac{\sum_{x\leq x_c^i} (x - x_c^i)^2 P_x^i(x)}{\sum_{x\leq x_c^i} P_x^i(x)} \\ u_{02}^{i+} = \frac{\sum_{y>y_c^i} (y - y_c^i)^2 P_y^i(y)}{\sum_{y>y_c^i} P_y^i(y)} \\ u_{02}^{i-} = \frac{\sum_{y\leq y_c^i} (y - y_c^i)^2 P_y^i(y)}{\sum_{y\leq y_c^i} P_y^i(y)} \end{array} \right. \quad (3-21)$$

$$\left\{ \begin{array}{l} x_c^i = \frac{\sum_x x P_x^i(x)}{\sum_x P_x^i(x)} \\ y_c^i = \frac{\sum_y y P_y^i(y)}{\sum_y P_y^i(y)} \end{array} \right. \quad (3-22)$$

$$\left\{ \begin{array}{l} x^{(i)}(x) = u(x)x^{(i)}(x) \\ y^{(i)}(x) = v(y)y^{(i)}(y) \end{array} \right. \quad (3-23)$$

3.2.3 Directional feature extraction

In the on-line character recognition, most of recognition methods are based on some sort of stroke matching technique. This usually involves finding which stroke of the input pattern corresponds to which stroke of the reference pattern, calculating the similarities between the input pattern and the reference pattern using stroke similarities. Recognition is accomplished by selecting the reference pattern having the greatest similarity to the input pattern. To keep high accuracy, input character pattern is required written with correct stroke numbers and by correct stroke orders. However, Japanese characters consist of many strokes and are written with varying stroke numbers and in varying stroke orders. So, these constraint make most systems inconvenient. Nowadays, methods permitting users to write characters with varying stroke numbers and in varying stroke orders have been developed. In recent years, the directional feature is extensively used in OCR and obtained better recognition rate than by on-line matching methods.

The implementation of direction feature extraction is various depending on the directional element decomposition, the sampling of feature values, the resolution of direction and feature plane, etc. Considering that the stroke segments of Japanese characters can be approximated into four orientations: horizontal, vertical, left-diagonal and right-diagonal, early works used to decompose the stroke (or contour) segments into these four orientations. Further, Liu etc proposed to decompose the stroke into eight, even 12 and 36 directions. Generally speaking, four and eight directional feature is widely used. Of course, decomposing the contour pixels into eight directions instead of four orientations (a pair of opposite directions merged into one orientation) significantly improved the recognition accuracy [75]. This is because separating the two sides of a stroke edge can better discriminate the parallel strokes.

3.2.4 Blurring and sampling

Each direction plane, with the standard size as the normalized image, need to be reduced to extract feature values of moderate dimensionality. A simple way is to partition the direction plane into a number of block zones and take the total or average value of each zone as a feature value. Partition of variable-size zones was proposed to overcome the non-uniform distribution of stroke density [76]. Overlapping blocks alleviate the effect of stroke-position variation on the boundary of blocks [77], yet a more effective way involves partitioning the plane into soft zones, which follows the principle of low-pass spatial filtering and sampling [78].

In implementation of blurring, the impulse response function (IRF) of spatial filter is approximated into a weighted window, also called a blurring mask. The IRF is often a Gaussian function given by Eq. (3-24):

$$h(x, y) = \frac{1}{2\pi\sigma_x^2} \exp\left(-\frac{x^2 + y^2}{2\sigma_x^2}\right) \quad (3-24)$$

According to the Sampling Theorem, the variance parameter σ_x relates to the sampling frequency (the reciprocal of sampling interval). On truncating the band-width of Gaussian filter, an empirical formula was given in [77]:

$$\sigma_x = \frac{\sqrt{2}t_x}{\pi} \quad (3-25)$$

where t_x is the sampling interval. At a location (x_0, y_0) of image $f(x, y)$, the convolution gives a sampled feature value

$$F(x_0, y_0) = \sum_x \sum_y f(x, y)h(x - x_0, y - y_0) \quad (3-26)$$

For ease of implementation, partition a direction plane into a mesh of equal-size blocks and set the sampling points to the center of each block. Assume to extract $K \times K$ values from a plane, the size of plane is set to $Kt_x \times Kt_x$. From N_d direction planes, the total number of extracted feature values is $N_d \times K^2$.

The extracted feature values are causal variables. Power transformation can make the density function of causal variables closer to Gaussian [79]. This helps improve the classification performance of statistical classifiers. Power transformation is also called variable transformation [80] or Box-Cox transformation [81]. Power 0.5 is employed to transform the variables or feature vector.

3.2.5 Dimensionality reduction

In order to reduce the computation complexity, we use fisher discriminant analysis (FDA) to reduce the dimensionality of feature vectors. In the process of FDA, we need between-class scatter covariance S_b and within-class scatter covariance S_w of training samples. Suppose there are C character classes ($\omega_1, \omega_2, \dots, \omega_C$) and the j -th class with N_j training samples. The total training samples is N . Then, S_w and S_b are defined as:

$$S_w = \sum_{j=1}^C \sum_{i=1}^{N_j} (x_j^i - \bar{x}_j)(x_j^i - \bar{x}_j)^T \quad (3-27)$$

$$S_b = \sum_{j=1}^C N_j (\bar{x}_j - \bar{x})(\bar{x}_j - \bar{x})^T \quad (3-28)$$

where $X = \{x_j^i\}$ ($j=1,2,\dots, C, i=1,2,\dots, N_j$) is set of samples with n -dimensions. $\bar{X}_j = \frac{1}{N_j} \sum_{i=1}^{N_j} x_j^i$

and $\bar{X} = \frac{1}{N} \sum_{i=1}^C \sum_{i=1}^{N_j} x_j^i$ are the mean vector of the j -th class and all classes, respectively.

Based on the Fisher discriminant criterion [101], the process of working out the transformation matrix is to find out the optimal ratio which makes the S_b as large as possible while making the S_w as small as possible, which is described as

$$W_{opt} = \arg \max_W \left| \frac{W^T S_b W}{W^T S_w W} \right| = [W_1 W_2 \dots W_m] \quad (3-29)$$

where $\{W_i | i=1,2,\dots,m\}$ are m n -dimensional eigenvectors of $S_w^{-1} S_b$ corresponding to the m largest eigenvalues. W_{opt} is the $n \times m$ matrix composed of the m n -dimensions eigenvectors. By the transformation matrix W_{opt} , we can reduce the dimensionality of feature vectors from n -dimensions to m -dimensions.

3.2.6 MQDF-based off-line character recognizer

The MQDF is the smoothed version of QDF, which performs Bayesian classification under the assumptions of multivariate Gaussian density for each class and equal a priori probabilities for all class. On an input pattern $X = (x_1, \dots, x_n)^T$, the QDF for class $\omega_i (i=1, \dots, M)$, has the form

$$g_0 = (X, w_i)^T \sum_i^{-1} (X - u_i) + \log |\Sigma_i| \quad (3-30)$$

where u_i and Σ_i denote the mean vector and the covariance matrix of class ω_i , respectively. The QDF is actually a distance metric in the sense that the class of minimum distance is assigned to the input pattern.

The QDF can be re-written in the form of eigenvectors and eigenvalues:

$$g_0(X, w_i) = \sum_{j=1}^d \frac{1}{\lambda_{ij}} [\varphi_{ij}(X - u_i)^T]^2 + \sum_{j=1}^d \log \lambda_{ij} \quad (3-31)$$

where $\lambda_{ij}, j = 1, 2, \dots, d$, denote the eigenvalues of class w_i sorted in decreasing order, and $\varphi_{ij}, j=1, 2, \dots, d$, are the corresponding eigenvectors. Replacing the minor eigen values with a larger constant, the modified quadratic discriminant function (MQDF2) is obtained as

$$g_2(X, w_i) = \sum_{j=1}^k \frac{1}{\lambda_{ij}} [\varphi_{ij}(X - u_i)^T]^2 + \frac{1}{\delta_i} D_c(X) + \sum_{j=1}^k \log \lambda_{ij} + (n - k) \log \lambda_{ij} \quad (3-32)$$

where λ_{ij} and φ_{ij} , $j = 1, 2, \dots, k$ denote the eigenvalue of class ω_i sorted in decreasing order and the corresponding eigenvectors respectively, k denotes the number of principal components and $D_c(X)$ is the square Euclidean distance in the complement subspace shown in Eq. (3-33), the parameter δ_i can be set as a class-independent constant as proposed by Kimura et al. [82] and $\text{tr}(\Sigma_i)$ denotes the trace of covariance.

$$D_c(X) = \|X - u_i\|^2 - \sum_{j=1}^k [\varphi_{ij}(X - u_i)^T]^2 \quad (3-33)$$

$$\delta_i = \frac{\text{tr}(\Sigma_i - \sum_{j=1}^k \lambda_{ij})}{(n - k)} = \frac{1}{n - k} \sum_{j=k+1}^n \lambda_{ij} \quad (3-34)$$

Then, the size of the off-line prototype dictionary \mathcal{S} is dependent on the data type of the parameters $u_i, \lambda_{ij}, \varphi_{ij}, \delta_i$ which are noted as $T_u, T_\lambda, T_\varphi, T_\delta$, respectively. In our system, T_u, T_λ and T_φ are integers with 16 bits; T_δ is a long integer with 32 bits. We can have the total size of the prototype dictionary given in Eq. (3-35), and N is the number of the character categories.

$$S = N \times \{n \times (T_u + k \times T_\varphi) + k \times T_\lambda + T_\delta\} \quad (3-35)$$

3.3 Recognizer Combination

The on-line and off-line character recognizers are combined by a linear function [33]. Suppose a character pattern x_i is recognized as a character class c_i by the on-line recognizer and off-line one with their similarity scores $f_{on}^{c_i}$ and $f_{off}^{c_i}$, respectively. Then, the confidence of the combined recognizer $f_{com}^{c_i}$ by the sum rule with class-independent linear combining parameters is given by the following formula:

$$f_{com}^{c_i} = \lambda_1 f_{on}^{c_i} + \lambda_2 f_{off}^{c_i} \quad (3-36)$$

where λ_1 and λ_2 are parameters. We use the minimum classification error (MCE) criterion to optimize the parameters, which will be described in chapter 5.

4. Orientation Free On-line Handwritten Text Recognition System

4.1 Introduction

This chapter describes an on-line handwritten Japanese text recognition system that is liberated from constraints on line direction and character orientation. The recognition system first separates freely written text into text line elements, second estimates the line direction and character orientation using the time sequence information of pen-tip coordinates, third hypothetically segment it into characters using geometric features and apply character recognition. The final step is to select the most plausible interpretation by evaluating the likelihood composed of character segmentation, character recognition, character pattern structure and context. The method can cope with a mixture of vertical, horizontal and skewed text lines with arbitrary character orientations. It is expected useful for tablet PC's, interactive electronic whiteboards and so on.

4.2 Line Direction and Character Orientation

Here, we define some terminologies. A stroke denotes a sequence of pen-tip coordinates sampled from pen down to pen up. An off-stroke is a vector from a preceding stroke to a succeeding stroke. Character orientation is used to specify the direction of character pattern from its top to bottom while line direction is used to designate the writing direction of a sequence of character patterns until it changes as shown in Fig.4-1. Although the line direction is the same as common sense, the character orientation might be the opposite from it. We define them in this way since they are consistent with pen-tip movement direction to write Japanese characters.

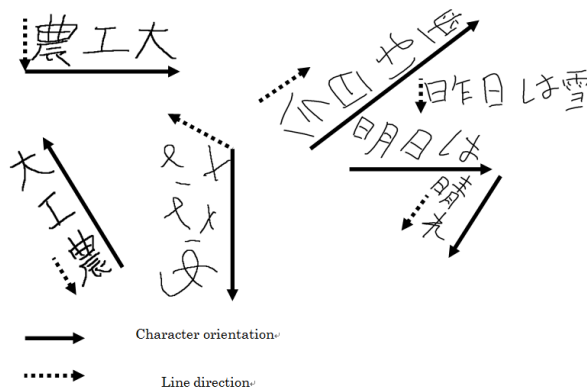


Figure4- 1 Character orientation and line direction.

An on-line handwritten document, or often called (digital) ink document is formed by text lines and line drawings, and all of them are formed by a sequence of strokes. In this paper, we focus on text but it is made of arbitrary line direction and character orientation.

A text line is a piece of text separated by new-line and large space and it is further divided into text line elements at the changing points of writing direction. Each text line element has its line direction as shown in Fig. 4-2. The line direction and the character orientation are independent.

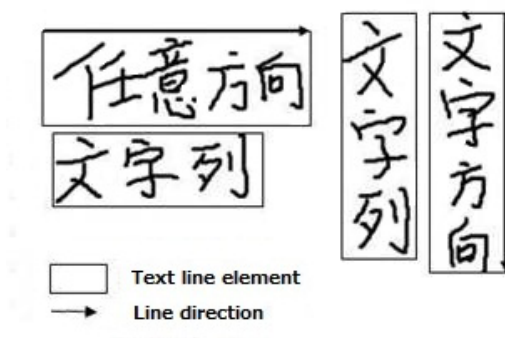


Figure4- 2 Text line element, Character orientation and line direction.

4.3 Flow of Recognition Process

The line-direction-free and character-orientation-free on-line handwritten Japanese text recognition system is composed of the steps shown in Fig. 4-3.

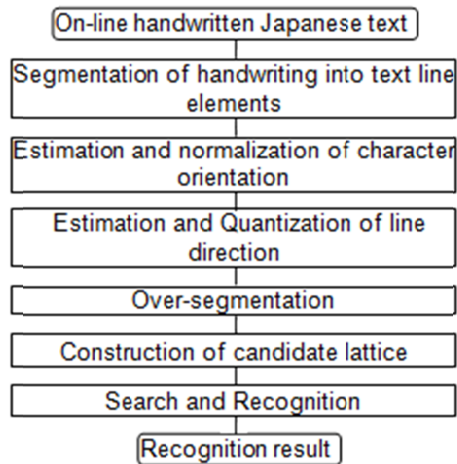


Figure4- 3 Recognition steps in the system.

It separates handwritten text of arbitrary character orientation and line direction into text line elements, estimates and normalizes character orientation, estimates and quantizes line direction, apply two-stage over-segmentation, constructs a segmentation-recognition candidate lattice (sr-lattice in short) and make the Viterbi search for the best path in the lattice to recognize handwriting. The path evaluation function combines the scores of character recognition, unary and binary geometric features as well as linguistic context where the weights for those factors are estimated by the genetic algorithm (GA) so as to optimize the holistic text recognition performance.

The following subsections describe them, especially updated processes in detail.

4.4 Segmentation of Handwriting into Text Line Elements

Text line segmentation is an important step in free-form on-line handwritten text recognition system. In our previous work[83], we have presented a solution to text line segmentation method, which based off-stroke(pen-up) distances and changing of writing directions based on the fact that off-strokes with in a text line are mostly shorter than those between text lines and the text lines are usually straight. However, this segmentation method does not perform reliably. [84] Propose a bottom-up method which is based on minimal spanning tree (MST) clustering of connected components for text line segmentation in free-form off-line handwritten Chinese documents. [85] Propose a approach to the detection of on-line handwritten text lines based on dynamic programming. The system tries to find a path between

two consecutive text lines based on a heuristic cost function that takes different criteria into account. [86] Over-segments the stroke sequence by DP with a cost function reflecting the confidence that a given set of strokes belongs to one word, and the text lines are grouped by merging pairs of stroke clusters in aggressive steps. In [87], an efficient text line segmentation method, is proposed and demonstrated to be more effective than most of existing methods.

The main idea of the method is employed for both temporal and spatial merge.

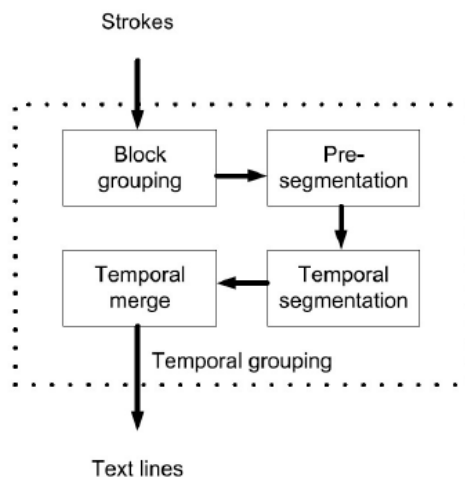


Figure4- 4 Flow chart of text line segmentation process(figure from [87]).

4.4.1 Line segmentation algorithm

Inspired by the work in [87], in this study, we propose a four steps algorithm for text line segmentation: (1) Block grouping. (2) Pre-segmentation. (3) Temporal segmentation. (4) Temporal merge.

(1) Block grouping

In order to alleviate the computation cost, consecutive strokes with small off-stroke distance are merged as blocks. The off-stroke distance feature has been used in [12].

It is calculated from the ending point of the preceding stroke to the starting point of the succeeding stroke. An off-stroke distance (OD) is defined as

$$OD = \frac{\sqrt{(x_i - x_{i+1})^2 + (y_i - y_{i+1})^2}}{acs * 0.3} \quad (4-1)$$

where acs is the average character size, which is estimated by measuring the longer side length

of the bounding box of each stroke, sorting the lengths of all the strokes and taking the average of the larger 1/3 of them. (x_i, y_i) and (x_{i+1}, y_{i+1}) are the coordinate of the ending point of the preceding stroke and the starting point of the succeeding stroke, respectively. If the off-stroke distance between two successive strokes is smaller than a threshold, they are marked as belonging to the same block.

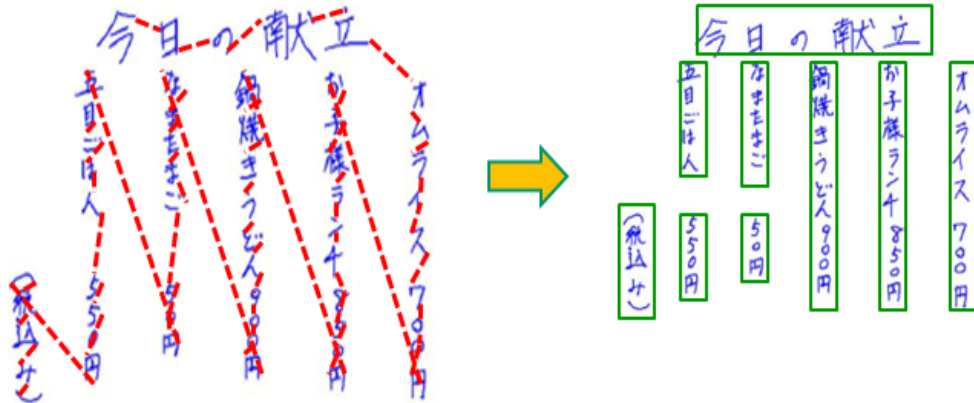


Figure4- 5 Line segmentation - block grouping.

(2) Pre-segmentation

For pre-segmentation, we first define the off-stroke between two consecutive blocks as the off-stroke between the last stroke of the preceding block and the first stroke of the succeeding block. Off-strokes within a text line are usually shorter than those between text lines. If the off-stroke between two consecutive text blocks is longer than a threshold, this off-stroke is regarded as a segmentation position. The threshold is empirically set as five times the average character size in our experiments to guarantee merging within-line blocks but risk over-merging multiple text lines with short off- strokes between them. On splitting the sequence of blocks at segmentation positions, each sub-sequence is to be split into text lines in succeeding temporal segmentation considering the linearity of stroke blocks.

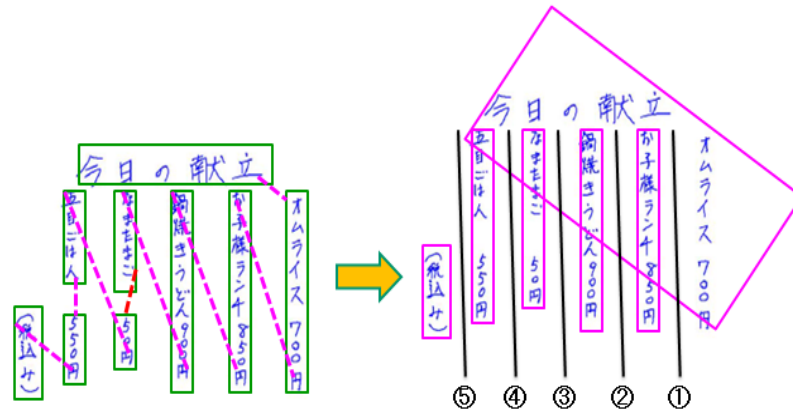


Figure4- 6 Line segmentation - pre-segmentation.

(3) Temporal segmentation

After pre-segmentation, each subsequence of stroke blocks (also called a text line) can be split into multiple text lines at internal off-strokes (candidate separation points). Since local information in the block sequence is not reliable enough to segment the sequence, we adopt a classification based method with text-level training which can evaluate the segmentation with a global objective function. To segment the sequence, each off-stroke between blocks can be taken as a candidate separation point, and the combination of all the candidate separation points form a candidate lattice, where each edge represents a candidate text line and a path from the start to the end represents a partitioning of text lines. The paths are evaluated using a trained discriminant function with the global features of the segmentation as inputs and the optimal path is obtained by beam search. At the end of this step, the text lines that have very small size and do not overlap with other lines.

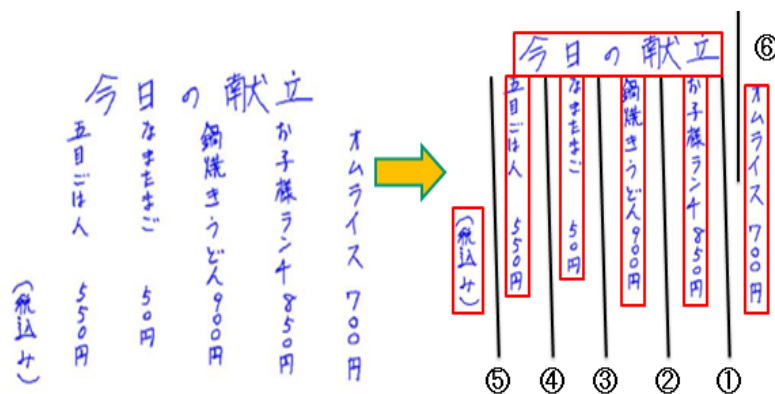
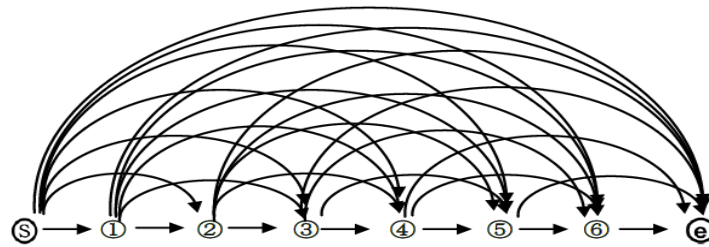
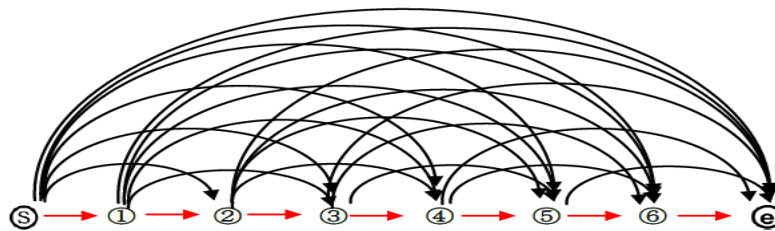


Figure4- 7 Line segmentation - temporal segmentation.



Segmentation point and segmentation candidate lattice



Temporal segmentation choose the best path

Figure4- 8 Segmentation candidate lattice of a text line string with six blocks(six straight edges). Each internal node corresponds to an off-stroke between blocks. The curved edges denote text lines comprising multiple blocks.

(4) Temporal merge

After stroke classification (prior to text line grouping), some text strokes are misclassified, which will split a text line into multiple ones. The temporal merge module is designed to correct such stroke classification errors and merge the over-segmented text lines. For this stage, an SVM classifier is trained to make the merge/non-merge decision for each hypothesis.

(5) Spatial merge

In on-line handwritten text, strokes are mostly written in character order, but there are still some delayed strokes, which are added to a former character after a later character of the text line is written. In temporal grouping of text lines, such delayed strokes cause two types of segmentation errors: the block (or short text line) of delayed strokes is embraced by a long text line. And the ends of two collinear text lines are close to each other but are temporally separated by delayed strokes. The spatial merge module is intended to merge such over-segmented text lines.

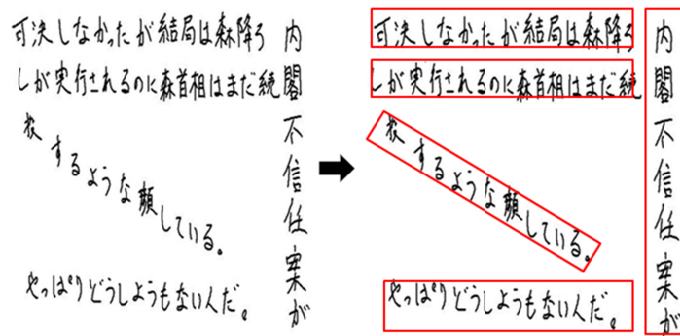


Figure4- 9 An example of text line segmentation.

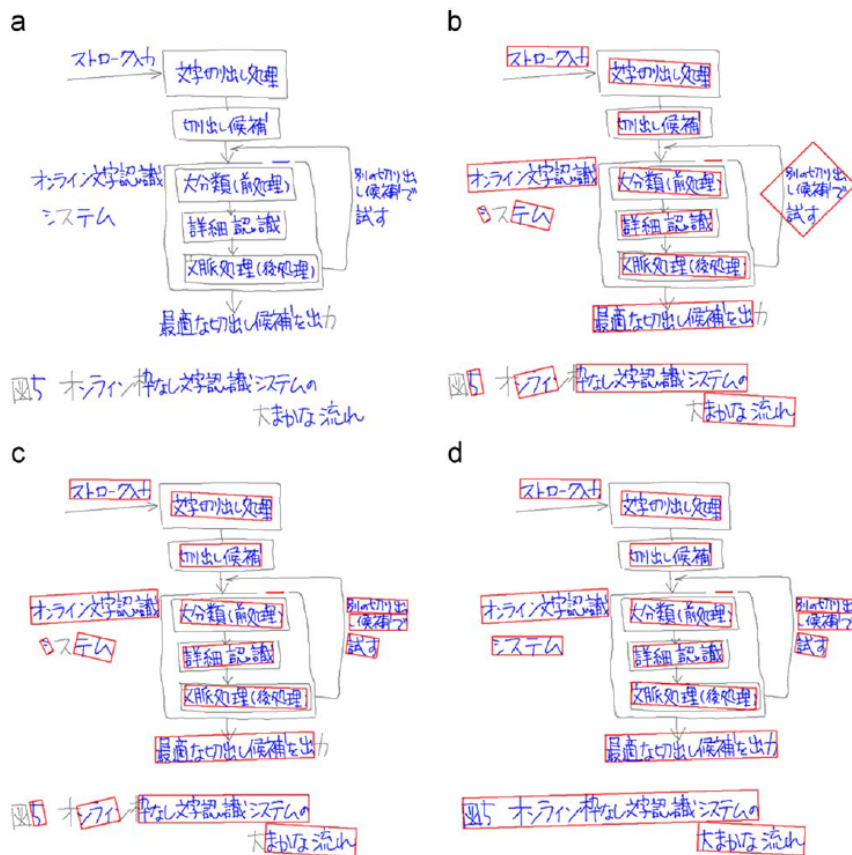


Figure4- 10 Grouping process. The grouping results are bounded with red rectangular boxes: (a) stroke classification results,(Blue: text stroke; gray: non-text strokes.) (b) pre-segmentation results, (c) temporal segmentation results, and (d)temporal merge results (figure from[87]).

4.5 Estimation and Normalization of Character

Orientation

This is made by the two steps as shown in Fig. 4-11. It produces multiple hypotheses and the succeeding recognition stages determine the best estimation.

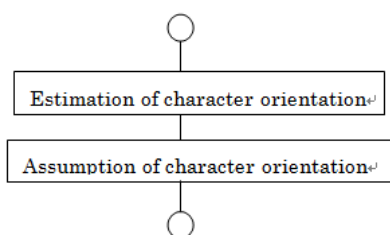


Figure4- 11 Flow processing of character orientation

4.5.1 Estimation of character orientation

When Japanese characters are written, principal pen movement within real strokes as shown in Fig. 4-12 is the same as the character orientation or $\pi/2$ counter clockwise to it. This is because Japanese characters, especially Kanji characters, are composed of downward and rightward strokes. Because of this, if we take the histogram of displacement direction of pen tip coordinates, we will see two peaks as shown in Fig.4-14. These peaks are not so stable if characters are few in a text line element, but they become more stable as the number of characters increases. Therefore, we can estimate the character orientation from the histogram of displacement direction for a text line element. Once, the character orientation is estimated, the text line element can be recognized by rotating characters until their orientation become downward. Let us assume the intensity of the histogram at the angle θ as $f(\theta)$. Then, compute $f(\theta) * f(\theta + \pi/2)$. This is to find the overlap between $f(\theta)$ and $f(\theta + \pi/2)$. If we can find a single and strong peak, this implies that the peak at θ and that of $\theta + \pi/2$ are notable and θ is the character orientation. In order to make the peak detection more robust, we take convolution of $f(\theta)$ and the Gauss function $g(\delta) = \exp(-\delta^2/\sigma^2)$ to blur the peak as shown in Fig. 4-13,4-14 so that it works for slanted characters that have rightward strokes with slightly upward inclination.

The system estimates the character orientation as 0, 30, 60 and so on, namely being quantized by 30 degrees, since is a tolerance of our character recognizer.

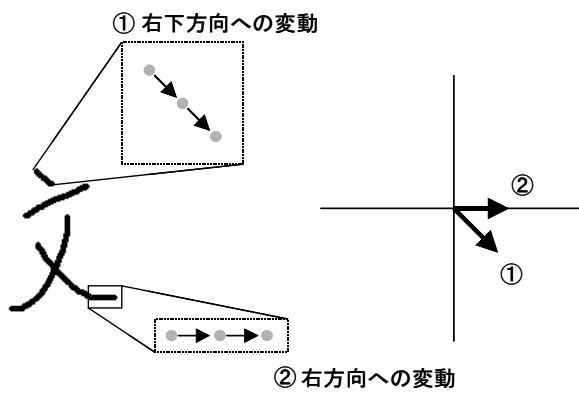


Figure4- 12 Pen movement

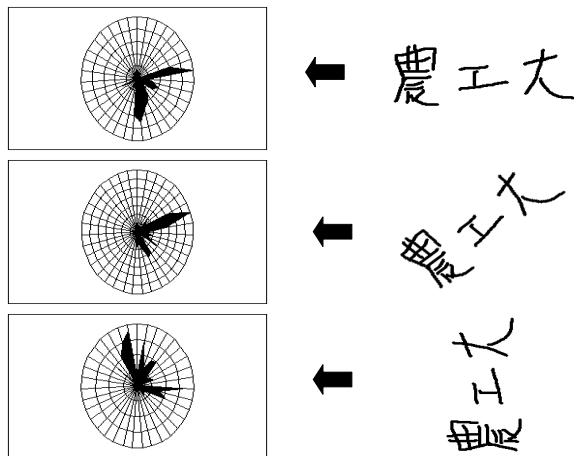


Figure4- 13 Two main peaks in pen movement direction.

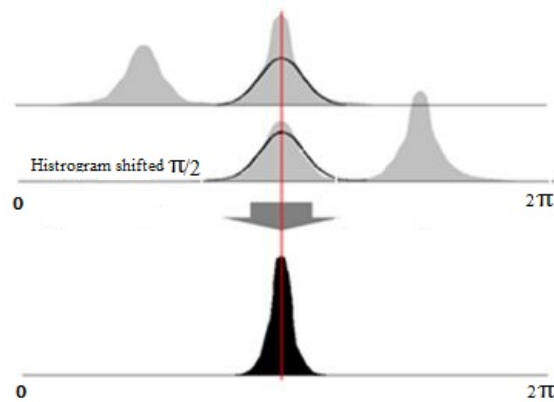


Figure4- 14 Estimation method of character orientation(figure form [11]).

4.5.2 Assumption of character orientation

This step assumes character orientation commonly appearing in relation to the line direction of a text line element. Assumed orientations are the same and opposite of the line direction, two perpendicular orientations to it and the four orientations (upward, downward, rightward and leftward) to the orientation of the input tablet as shown in Fig. 4-15. We can expand the assumption more than these orientations or restrict them when the character orientation has the strong sign or it is confined from applications.

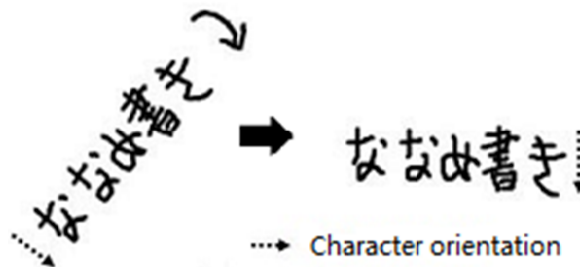


Figure4- 15 Normalization of character orientation.

4.6 Estimation and Quantization of Line Direction

Here again, we succeed the original process. The direction of a text line element can be estimated by the width and height of the bounding box and the slant degree from the start point to the end point of the text line element.

Then, the direction of each text line element is quantized into 4 directions, rightward (*R*), leftward (*L*), upward (*U*) and downward (*D*) as shown in Fig. 4-16. Even though there are some text lines near to or just at the line direction boundaries (45, -45, 135,-135), we can decide their

line directions without bringing any problem. For instance, the text line “Kinou ha yuki” in Fig. 4-1, can be detected into leftward or upward. No matter it is leftward or downward our system can work correctly, because we have trained the data for every line direction.

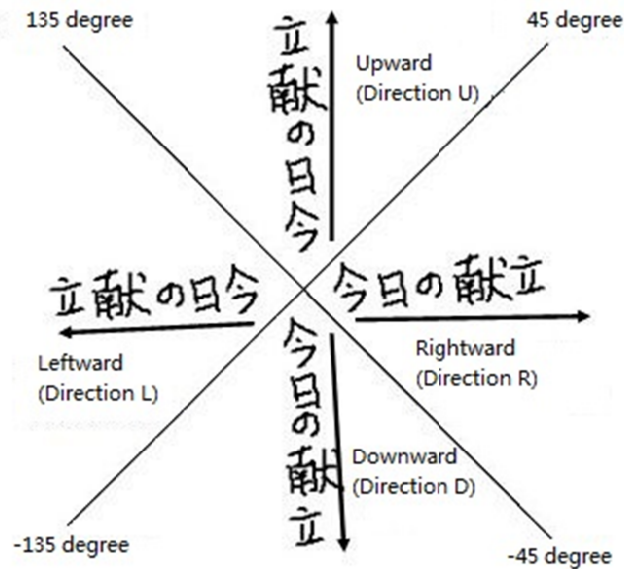


Figure4- 16 Quantization of line direction.

4.7 Over-Segmentation

This is the main part of our update. It is composed of two stages and the latter stage employs dimensionality reduction.

Each off-stroke is classified into three classes: segmentation point (*SP*), non-segmentation point (*NSP*) or undecided point (*UP*) according to geometric features. A segmentation point separates two characters at the off-stroke while a non-segmentation point indicates the off-stroke is within a character. Off-strokes with low classification confidence are undecided points. A sequence of consecutive strokes between two adjacent segmentation/undecided points is a primitive segment, and one or more consecutive primitive segments form a candidate character pattern.

In order to minimize the misclassification we follow the two-stage classification scheme [8]. First, we use two geometric features to classify off-strokes into hypothetical segmentation points (*SP* and *UP*) or non-segmentation points (*NSP*). Then, we apply SVM to further classify hypothetical segmentation points into *SP* or *UP*. The process is detailed in the following subsections.

4.7.1 Stage 1: Classification by two features

A text line element is hypothetically segmented depending on its quantized direction. We generate hypothetical segmentation points based on features of each off-stroke: the distance feature f_d , (horizontal distance when the line direction is R or L while vertical distance when the line direction is D or U , and the intersecting length feature f_i that calculates the overall intersecting length between the group of all strokes preceding the off-stroke and the group of all succeeding strokes. These two features were proposed for ordinary horizontal text line [8]. Here we extend them for any other line directions.

BBp_all: Bounding box of all the preceding strokes;
BBs_all: Bounding box of all the succeeding strokes;
acs: Average character size;
DBx: Distance between *BBp_all* and *BBs_all* in X-direction;
DBy: Distance between *BBp_all* and *BBs_all* in Y-direction;

```

if(Line direction = R)
  DBx = X coordinate of the left position of BBs_all -
  X coordinate of the right position of BBp_all;
if(Line direction = L)
  DBx = X coordinate of the left position of BBp_all -
  X coordinate of the right position of BBs_all;

if(Line direction = D)
  DBy = Y coordinate of the top position of BBp_all -
  Y coordinate of the right position of BBs_all;
if(Line direction = U)
  DBy = Y coordinate of the top position of BBs_all -
  Y coordinate of the bottom position of BBp_all;

```

Then, f_d is extracted as follows:

```

if(Line direction = R or L)  $f_d = DBx / acs$ 
else  $f_d = DBy / acs$ 

```

Figure4- 17 A quasi-program to obtain the feature f_d

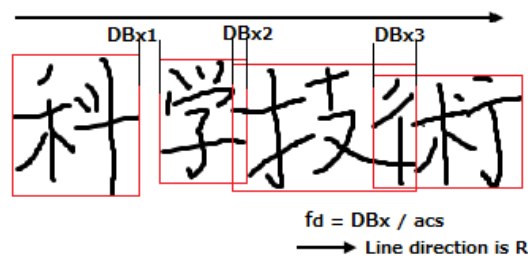


Figure4- 18 Distance feature f_d for line direction R.

The algorithm to obtain the distance feature f_d , is shown in Fig. 4-17 with the definitions of necessary terms. Fig. 4-18 shows an example.

The term *acs* is the average character size, which is estimated by measuring the longer side length of the bounding box of each stroke for each text line element, sorting the lengths of all the strokes and taking the average of the larger 1/3 of them.

On the other hand, the intersecting length feature f_i is obtained by the algorithm shown in Fig. 4-19 and it is depicted in Fig. 4-20.

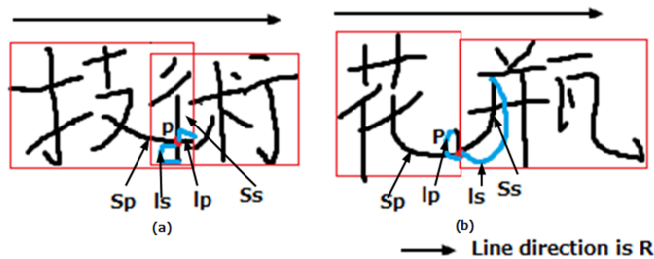
S_{p_all} : Group of all the preceding strokes;
 S_{s_all} : Group of all the succeeding strokes;
 s_p : a stroke in S_{p_all} ;
 s_s : a stroke in S_{s_all} ;
 l_p, l_s : Length from the intersecting point;
 $l(s_p, s_s)$: Intersecting length between s_p and s_s ;

if(s_p and s_s are intersecting at a point p)
 if(Line direction = R)
 l_p = partial length of s_p from p to the right end of s_p ;
 l_s = partial length of s_s from p to the left end of s_s ;
 if(Line direction = L)
 l_p = partial length of s_p from p to the left end of s_p ;
 l_s = partial length of s_s from p to the right end of s_s ;
 if(Line direction = D)
 l_p = partial length of s_p from p to the bottom end of s_p ;
 l_s = partial length of s_s from p to the top end of s_s ;
 if(Line direction = U)
 l_p = partial length of s_p from p to the top end of s_p ;
 l_s = partial length of s_s from p to the bottom end of s_s ;
 else $l_p = 0, l_s = 0$;
 $l(s_p, s_s) = -\min(l_p, l_s) / \max(\text{length of } s_p, \text{length of } s_s)$;
 L_{sum} : Accumulated intersecting length;

$$L_{sum} = \sum_{s_p \in S_{p_all}} \sum_{s_s \in S_{s_all}} l(s_p, s_s)$$
 f_i : Overall intersecting length for an off-stroke

$$f_i = \begin{cases} L_{sum}, & \text{if } L_{sum} < 0 \\ 0, & \text{otherwise} \end{cases}$$

Figure4- 19 A quasi-program to obtain the feature f_i



(a)for strokes intersecting at point P (b) for strokes contact at point P.

Figure4- 20 Features to obtain f_i for line direction R.

When the distance feature is positive at an off-stroke, we treat it as a hypothetical segmentation point. When it is less than or equal to 0, we consider the distribution of their values for true segmentation points.

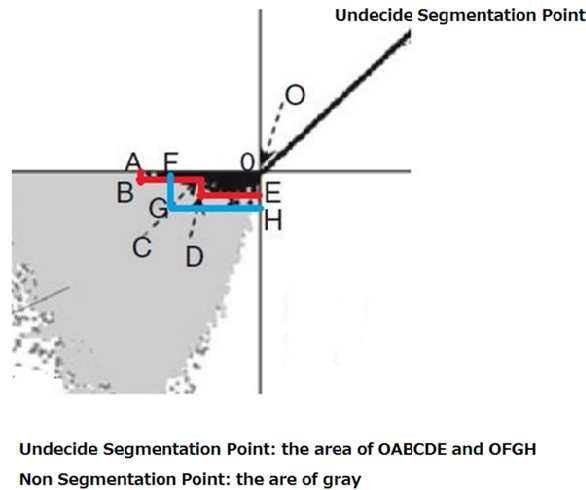


Figure4- 21 Setting thresholds for hypothetical segmentation (figure from [8]).

Fig. 4-21 shows the distribution of the distance feature and the intersecting length feature of off-strokes in a set of training string samples when the distance feature is less than or equal to 0. We can see that segmentation points and non-segmentation points are separated to a large extent by these two features. We classify off-strokes as hypothetical segmentation points if their values are in the area of OABCDE, and as non-segmentation points otherwise. After this, we modify the classified non-segmentation points between two successive hypothetical segmentation points in the area of OFGH as hypothetical segmentation points, if the width between the two successive hypothetical segmentation points divided by acs is greater than a threshold. We defined these areas heuristically by plotting f_i and f_d from segmentation samples.

4.7.2 Dimensionality reduction (PCA)

We apply Principal Component Analysis (PCA) to reduce geometric features for over-segmentation since we can adjust the memory size and recognition speed without losing recognition and segmentation accuracy significantly. We then use a SVM classifier to classify the hypothetical segmentation points on the reduced features.

Given N samples \mathbf{x}_i ($i \leq N$) of n -dimensional feature vectors, the total scatter covariance is defined as follows:

$$\mathbf{S}_T = \sum_{i=1}^N (\mathbf{x}_i - \mathbf{u})(\mathbf{x}_i - \mathbf{u})^T \quad (4-2)$$

where \mathbf{u} is the mean vector of all samples. We can obtain the optimal matrix \mathbf{W}_{opt} , according to the following formula:

$$\mathbf{W}_{opt} = \arg \max_{\mathbf{W}} |\mathbf{W}^T \mathbf{S}_T \mathbf{W}| = |\mathbf{W}_1 \mathbf{W}_2 \dots \mathbf{W}_m| \quad (4-3)$$

where $\{\mathbf{W}_i; i=1, 2, \dots, m\}$ are m n -dimensional eigenvectors of \mathbf{S}_T corresponding to the m largest eigenvalues. Then we can reduce the extracted 21 geometric features according to the following formula:

$$\mathbf{y}_i = \mathbf{W}_{opt}^T \mathbf{x}_i (\mathbf{x}_i \in R^n, \mathbf{y}_i \in R^m) \quad (4-4)$$

4.7.3 SVM classification

A handwritten text pattern is composed of many characters with a sequence of strokes, In Japanese, different kinds and complexities of characters: Kanji, Hiragana, Katakana, numeric characters and others are mixed. An input text pattern should be correctly segmented into each character as far as possible. It is difficult, however, due to the facts that spaces between characters are not obvious, many characters include multiple radicals with internal gaps and some characters are connected in writing. To solve these problems, a text pattern is over-segmented into a sequence of primitive segments so as to segment true segmentation points surely but may segment single character patterns into pieces, which could be combined in the later text recognition stage. Zhu et al. employ two-stage segmentation scheme [8]. In the first stage, each off-stroke (a vector from the last point of a previous stroke to the first point of the next stroke) is classified into non-segmentation point (*NSP*) and hypothetical one based on geometric features. Then, in the second stage, each hypothetical point is classified into segmentation point (*SP*) and undecided point (*UP*) using SVM model according to 20-dimensional features extracted from an off-stroke, where a *SP* separates two characters at the off-stroke, an *NSP* indicates the off-stroke is within a character and a *UP* is interpreted either as a *SP* or an *NSP*. When it is interpreted as a *SP*, it is used to extract candidate character patterns beside it with nearest neighbor *SPs* or *UPs* interpreted as *SPs*. When it is interpreted as an *NSP*, it is considered within a character pattern and does not play a role for segmentation. We call a

sequence of strokes delimited by *SP* or *UP* as a primitive segment.

In a character-position-free handwritten text pattern, however, spaces between characters are very unstable. We can't directly use the conventional handwritten text recognition model. The first stage in the above-mentioned recognizer may combine two characters since the space between them disappears. Therefore, we remove the first stage and only employ the second stage.

The next concern is the classification of off-strokes into *NSP*, *SP* or *UP*. We may follow this scheme or change the scheme. In this paper, we compare two segmentation methods. The first one is the conventional method to classify off-strokes into *NSP*, *SP* or *UP* although all the parameters and thresholds are retrained according to the new training patterns. We call this method “candidate segmentation method”. On the other hand, we set every off-stroke as *UP* in the alternative method although we employ the output of SVM model in the text recognition stage. We call it “undecided segmentation method”.

Namely, the first method classifies off-strokes into *NSP*, *SP* or *UP*, but the second method treats every off-stroke as *UP*. Both of the two methods, however, transform the output of SVM to segmentation probability value. Moreover, the segmentation probability value is combined into the optimal path evaluation in candidate segmentation-recognition paths.

In segmentation methods, we need to extract more geometric features from an off-stroke in order to enhance the reliability of over-segmentation. Through investigation into related literatures, we employ all the useful geometric features proposed so far, i.e.56-dimensional features, to train SVM model. The detail will be described in the next subsection.

SVM Model

Support vector machines (SVMs) developed from statistical learning theory [90] for pattern recognition, have been successful applied to the handwriting segmentation task. Sun et al. [84] compared different supervised classifiers for classifying gaps between pieces of handwritten text to inter-word and intra-word classes, and found that SVMs outperform the other classifiers. Zhu et al. [12] employed SVM to determine segmentation point candidates for improving on-line freely written Japanese text recognition. Moreover, they showed that the character recognition rate by SVM-based segmentation are better than that by the three-layers neural network, although SVM method takes more training time than the neural network. Harbi et al. [91] also employed a linear kernel-based SVM classifier with temporal and spatial features for clock drawing segmentation, and showed this method outperforms the current state-of-the-art method on two collected datasets.

As for the character-position-free on-line handwritten text segmentation, we continue employ SVM classifier to segment each off-stroke with more geometric features.

Support Vector Machine (SVM)

Suppose we are given a training set $D_T = \{(x_i, y_i) | i = 1, \dots, N\} \in (X \times Y)^N$, where $x_i \in X = R^n$ stands for the feature vector of a training pattern i , and $y_i \in Y = \{-1, 1\}$ is an associated class label of a training pattern i , N is the number of training patterns, respectively.

Then, by mapping from the space of R^n to the high dimension space H :

$$\varphi: \begin{matrix} X=R^n \rightarrow H \\ x \rightarrow \varphi(x) \end{matrix} \quad (4-5)$$

D_T is mapped as:

$$D'_T = \{(x_i, y_i) | i = 1, \dots, N\} = \{(\varphi(x_i), y_i) | i = 1, \dots, N\} \quad (4-6)$$

The key idea of SVM is to learn the parameters of the hyper plane in space H that has maximum margin to classify two classes on training set.

To find the hyper plane $w \cdot x_i + b = 0$, it can be translated into the following optimization problem:

$$\begin{cases} \min: \frac{1}{2} \|w\|^2 + C \sum_{i=1}^n \xi_i \\ \text{s. t. } \xi_i \geq 0, y_i(w \cdot x_i + b) \geq 1 - \xi_i \end{cases} \quad (4-7)$$

where $\|w\|$ stands for the maximum margin, ξ_i is the learning error of a training pattern i , C is the trade-off between learning error and maximum margin, respectively.

Then, the feature vectors are mapped into an alternative space choosing kernel function $k(x_i, x_j) = \varphi(x_i) \cdot \varphi(x_j)$ for nonlinear discrimination. Consequently, it leads to the following quadratic optimization problem:

$$\left\{ \begin{array}{l} \min: W(\alpha) = \frac{1}{2} \sum_{i=1}^N \sum_{j=1}^N y_i y_j \alpha_i \alpha_j K(x_i, x_j) + \sum_{i=1}^N \alpha_i \\ \text{s. t: } \sum_{i=1}^N y_i \alpha_i = 0, \forall i: 0 \leq \alpha_i \leq C \end{array} \right. \quad (4-8)$$

where α is a vector of N variables and each element α_i corresponds to a training pattern (x_i, y_i) .

The solution of the optimization problem as shown in Eq. (4-8) is to find a vector α^* to let $W(\alpha)$ is the minimum and the constraints are fulfilled. The classification of an unknown pattern x_i made based on the sign of the following function $f(x)$, where SV stands for support vector as shown in Fig. 4-22.

$$f(x) = \sum_{i:SV} \alpha_i^* y_i K(x, x_i) + b^* \quad (4-9)$$

In this thesis, we set the target value of segmentation points as 1, and that of non-segmentation points as -1. We use SVM_{light}[92] to obtain the separating hyper plane by solving this optimization problem as shown in Eq. (4-9) on training patterns. This software efficiently solves classification problem with many thousand support vectors, and converge with fast optimization algorithm.

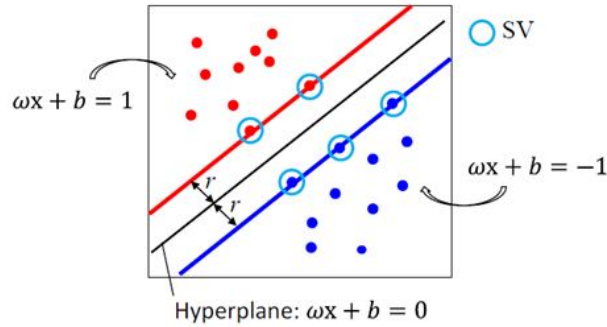


Figure4- 22 Example of Support vectors (figure from [92]).

Features for SVM

An off-stroke is evaluated by SVM model. We cover all the useful geometric features from the literature [54] and [12], and extract 56 features from each off-stroke. Table4-1 shows the

features and table 2 shows terms to derive the 56 features. Most of the features are normalized by an average character size (acs). The average character size is estimated by measuring the length of the longer side of the bounding box for each stroke, sorting the lengths from all the strokes and taking the average of the larger 1/3 of them.

We just use order information for the on-line character recognition engine, but we do not use time information. Time information can be used to separate characters intentionally, but it causes mis-segmentation when a user stops writing in a character pattern. In automobile environment and even in ordinary environment, a user may stop writing or resume writing.

Table4- 1 Terms to obtain 21 features.

Symbol	The definition of the symbol
B_{bp}	Bounding box of the immediately preceding stroke
B_{bs}	Bounding box of the immediately succeeding stroke
D_{bx}	Distance between B_{bp} and B_{bs} to x-axis If (line direction = L) D_{bx} = X coordinate of the left position of B_{bp} - X coordinate of the right position of B_{bs} else D_{bx} = X coordinate of the left position of B_{bs} - X coordinate of the right position of B_{bp}
D_{by}	Distance between B_{bp} and B_{bs} to y-axis If (line direction = D) D_{by} = Y coordinate of the top position of B_{bp} - Y coordinate of the bottom position of B_{bs} else D_{by} = Y coordinate of the top position of B_{bs} - Y coordinate of the bottom position of B_{bp}
O_b	Overlap area between B_{bp} and B_{bs}
D_{bsx}	Distance between centers of B_{bp} and B_{bs} to x-axis D_{bsx} = X coordinate of the center of B_{bs} - X coordinate of the center of B_{bp}
D_{bsy}	Distance between centers of B_{bp} and B_{bs} to y-axis D_{bsy} = Y coordinate of the center of B_{bs} - Y coordinate of the center of B_{bp}
D_{bs}	Absolute distance of centers of B_{bp} and B_{bs}
D_{fb}	Difference between B_{bp_all} and B_{bs} If (Line direction = R or L) D_{fb} = abs(Y coordinate of the top position of B_{bp_all} - Y coordinate of the top position of B_{bs}) else D_{fb} = abs(X coordinate of the top position of B_{bp_all} - X coordinate of the top position of B_{bs})

Table4- 2 21 features for over-segmentation extracted from each off-stroke.

Symbol	The definition of the symbol
$f1$	Time lapse of the off-stroke
$f2$	D_{bx}/acs
$f3$	D_{by}/acs
$f4$	Overlap area between B_{bp_all} and $B_{bp_all}/(acs)^2$
$f5$	$D_{bx} / \text{width of } B_{bp}$
$f6$	$D_{bx} / \text{width of } B_{bs}$
$f7$	D_{by} / acs
$f8$	$O_b / (\text{width} * \text{height of } B_{bp})$
$f9$	$O_b / (\text{width} * \text{height of } B_{bs})$
$f10$	D_{bsx} / acs
$f11$	$O_b / (\text{width} * \text{height of } B_{bp})$
$f12$	$O_b / (\text{width} * \text{height of } B_{bs})$
$f13$	$O_b / (acs)^2$
$f14$	D_{bsx} / acs
$f15$	D_{bsy} / acs
$f16$	D_{bs} / acs
$f17$	D_{fb} / acs
$f18$	Length of the off-stroke / acs
$f19$	Sine value of the off-stroke
$f20$	Cosine value of the off-stroke
$f21$	If(Line direction= R or L) $(D_{Bx} / acs) / \text{the maximum } (D_{Bx} / acs)$ in text else $(D_{By} / acs) / \text{the maximum}(D_{By} / acs)$ in text

4.8 Construction of SR-Lattice

Each candidate character pattern is associated with a number of candidate classes with

confidence scores by character recognition. All possible segmentations and recognition candidate classes are represented by a sr-lattice as shown in Fig. 4-23, where each arc denotes a candidate segmentation point and each node denotes a character class assigned to a candidate character pattern.

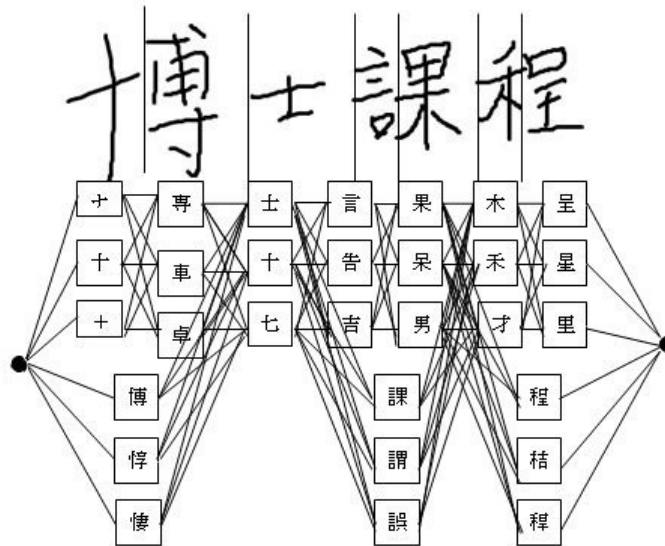


Figure4- 23 Segmentation-recognition candidate lattice.

4.9 Search and Recognition

We evaluate the lattice paths according to the path evaluation criterion first proposed by Zhu et al. [8] and formulated by Gao et al. [93] that combines the scores of character pattern size, inner gap, character recognition, single-character position, pair-character position, candidate segmentation point and linguistic context with weighting parameters estimated by GA. The optimal path can be found by the Viterbi search.

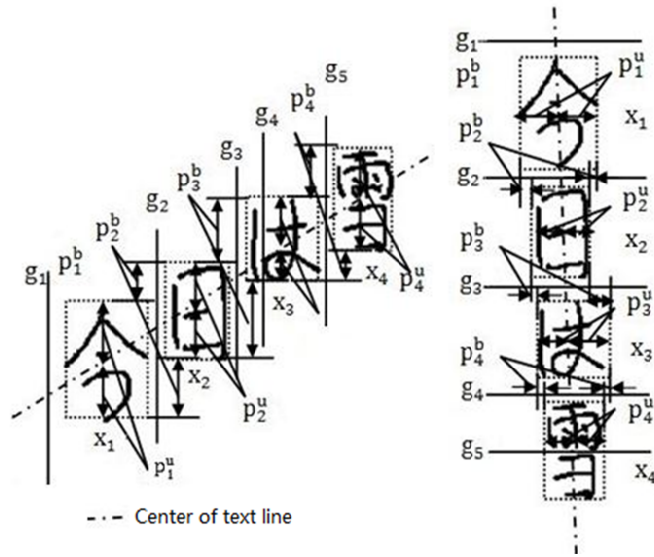
Denote $X = x_1 \dots x_m$ as the successive candidate character patterns of one path, and every candidate character pattern x_i is assigned candidate class C_i . Then, $f(X, C)$ is the score of the path (X, C) where $C = C_1 \dots C_m$. The criterion for path evaluation is expressed as:

$$f(X, C) = \sum_{i=1}^n \left(\begin{aligned} &(\lambda_{11} + \lambda_{12}(k_i - 1))\log P(C_i|C_{i-1}, C_i|C_{i-2}) + \\ &(\lambda_{21} + \lambda_{22}(k_i - 1))\log P(b_i|C_i) + \\ &(\lambda_{31} + \lambda_{32}(k_i - 1))\log P(q_i|C_i) + \\ &(\lambda_{41} + \lambda_{42}(k_i - 1))\log P(x_i|C_i) + \\ &(\lambda_{51} + \lambda_{52}(k_i - 1))\log P(p_i^u|C_i) + \\ &(\lambda_{61} + \lambda_{62}(k_i - 1))\log P(p_i^b|C_i, C_{i-1}) + \\ &\lambda_{71}\log P(g_{ij}|S_b) + \\ &\lambda_{72} \sum_{j=j_i+1}^{j_i+k_i-1} \log P(g_j|S_w) \end{aligned} \right) + \lambda n \quad (4-10)$$

where $b_i, q_i, x_i, p_i^u, p_i^b, g_{ij}$ and g_i are geometric features extracted whose details are explained below. The coefficients, $\lambda h_1, \lambda h_2$ ($h=1, \dots, 7$) and λ are weighting parameters, which are adjusted using GA to optimize the string recognition performance on a training dataset.

We start from describing p_i^u and p_i^b since they are extended for line-direction-free and character-orientation-free recognition. We compute the center line of each text line using a linear regression line that approximates the centers of the bounding boxes of the primitive segments for the text line. The term p_i^b consists of two values. For line direction R or L, it is composed of vertical distances from the center line to the top and bottom of the bounding box. For line direction U or D, it is composed of horizontal distances to the left and right of the bounding box. They are shown in Fig. 4-24.

The term p_i^b is composed of two values. For line direction R or L, it is composed of a vertical distance between the upper bounds and another vertical distance between the lower bounds of two adjacent candidate character patterns in a text line. For line direction U or D, it is composed of a horizontal distance between the left bounds and another horizontal distance between the right bounds of two adjacent candidate character patterns in a text line. They are shown in Fig. 4-24.



(a) for line direction R and L . (b) for line direction U and D .

Figure4- 24 Geometric features.

The term b_i is composed of the height and width of the bounding box of a candidate character pattern.

The term q_i is an inner-gap vector in a candidate character pattern, which is obtained by projecting the character pattern into the horizontal and vertical directions, splitting each of their histograms into three slices, finding a gap or gaps in each slice and summing total lengths of gaps as shown in Fig. 4-25.

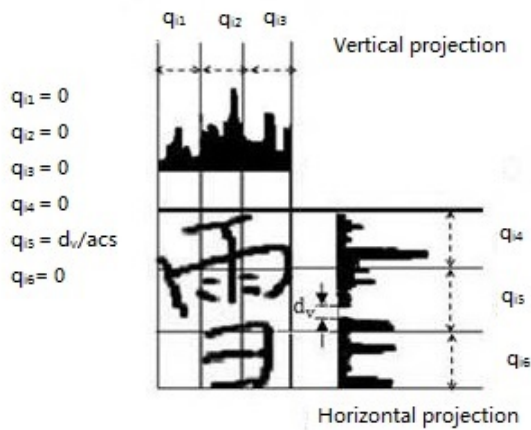


Figure4- 25 Inner gap feature within a character pattern.

The term g_{ji} and the term g_i comprise multiple features measuring the relationship between two primitive segments adjacent to a candidate segmentation point [93].

The values of geometric features $b_i, q_i, p_i^u, p_i^b, g_{ji}$ and g_i are normalized with respect to the average character size acs for scaling invariance. Several geometric features are shown in Fig. 4-24, 4-25.

The term x_i denotes features for a candidate character pattern x_i .

Then, the probabilities are as follows. $P(C_i/C_{i-1}, C_{i-2})$ is the tri-gram context probability. The probabilities $P(b_i/C_i), P(q_i/C_i), P(p_i^u/C_i)$ and $P(p_i^b/C_{i-1}, C_i)$ are assumed to be normal distributions and model their logarithms by a quadratic discriminant function (QDF), which can be trained by training patterns. $P(x_i/C_i)$ is evaluated by the character recognizer that combining the scores of the on-line and off-line recognizers [18]. $P(g_{ji} | S_b)$ is the probability that spacing between character patterns (S_b) appears as g_{ji} and $P(g_j | S_w)$ is the probability that spacing within a character pattern (S_w) appears as g_j . As the result of over-segmentation in 3.4, UP is interpreted as either S_b or S_w in a sr-lattice. It is treated as S_b when it is between character patterns and S_w otherwise. SP is always treated as S_b and NSP is always treated as S_w . The probabilities $P(g_i | S_b)$ and $P(g_i | S_w)$ are approximated by the SVM classifier[93].

4.10 Optimization of Parameters

We train all the weighting parameters $\lambda h_1, \lambda h_2$ ($h=1\sim 7$) and λ in Eq. (5-7). by the minimum classification error (MCE) criterion [40] or the genetic algorithm (GA), using training data of character-position-free text patterns to maximize the recognition rate on this training data.

4.10.1 MCE

Liu et al. [94] have applied this criterion on handwritten numeral string recognition to improve recognition performance.

In the character-position-free handwritten text recognition, the weighting parameters Λ are trained on a training set $D=\{X_i, C_i | i=1, \dots, N\}$, where C_i denotes the ground-truth text class label of a training sample X_i , and N is the number of training samples. Each class C is assigned a discriminant score $g(X_i, C, \Lambda)$. Following Juang et al. [49], the misclassification measure on a training sample from class C_i is given by:

$$d(X^i, C^i, \Lambda) = -g(X^i, C^i, \Lambda) + \log\left(\frac{1}{N-1} \sum_{c \neq c^i} e^{-\eta g(X^i, c, \Lambda)}\right) \quad (4-11)$$

where η is a positive number. When $\eta \rightarrow \infty$,

$$d(X^i, C^i, \Lambda) = -g(X^i, C^i, \Lambda) + g(X^i, \bar{C}^i, \Lambda) \quad (4-12)$$

where \bar{C}^i is the class label with the highest discriminant score in the closest rival class, namely,

$$g(X^i, \bar{C}^i, \Lambda) = \max_{c \neq c^i} g(X^i, c, \Lambda) \quad (4-13)$$

The loss of misclassification using sigmoid function is computed by,

$$l(X^i, C^i, \Lambda) = \frac{1}{1 + e^{-\xi d(X^i, C^i, \Lambda)}} \quad (4-14)$$

where ξ is a parameter. Then, the loss of misclassification based on training set is defined as:

$$L(\Lambda, D) = \frac{1}{N} \sum_{i=1}^N l(X^i, C^i, \Lambda) \quad (4-15)$$

We use the stochastic gradient descent [50] to learn the optimal parameters in Eq. (4-15). The parameters are updated on each training sample by

$$\Lambda(t+1) = \Lambda(t) - \varepsilon(t) \cup \nabla l(X^i, C^i, \Lambda) | \Lambda = \Lambda(t) \quad (4-16)$$

where $\Lambda(t)$ denotes the parameters on time t , $\varepsilon(t)$ is the learning step, U is related to the inverse of Hessian matrix and is usually approximated to be diagonal.

As for the character-position-free handwritten text recognition, MCE is to find the optimal parameters in Eq. (4-15) by minimizing difference between the scores of the most confusing text class and that of the correct one. The discriminant function is the path evaluation criterion defined in Eq. (4-9). The rival segmentation-recognition path, which is the most confusable one with the correct one, is obtained by beam search. Assume the discriminant functions f_c and f_r for the correct path and rival one, respectively. The parameters are updated iteratively by:

$$\Lambda(t + 1) = \Lambda(t) - \varepsilon(t)\xi l(X^i, C^i, \Lambda(t)) (1 - l(X^i, C^i, \Lambda(t)))(f_r - f_c) \quad (4-17)$$

4.10.2 GA

Zhu et al. [8] have reported that the GA-based parameter optimization method yields better recognition performance than MCE-based method for on-line handwritten Japanese text recognition. The GA-based method, however, takes more training times than MCE.

The parameters are estimated by a GA on the training text patterns as follows:

Step1 (initialization): Initialize N chromosomes with random values from 0 to 1, average fitness of the N chromosomes f_{old} as 0 and time t as 1.

Step2 (crossover): Select two chromosomes at random from N chromosomes. Cross the elements between two random positions to produce two new chromosomes. Repeat until obtaining M new chromosomes.

Step3 (mutation): Change each element of $N+M$ chromosomes with a random value from -1 to 1 at a probability P_{mut} .

Step4 (fitness evaluation): Evaluate fitness in terms of the recognition rate on training data with the weight values encoded in each chromosome.

Step5 (selection): Decide the roulette probability of each chromosome according to its fitness. First select two chromosomes with the highest fitness, and then select chromosomes using the roulette until obtaining N new chromosomes. Replace the old N chromosomes with the new ones.

Step6 (iteration): Obtain the average fitness of the new N chromosomes f_{new} . If $(f_{new} - f_{old} < \text{threshold})$ occurs n_{stop} times or $t > T$, return the chromosome of the highest fitness. Otherwise, set f_{new} to f_{old} , increment t , and go to step 2.

For evaluating the fitness of a chromosome, each training sample is searched for the optimal path evaluated using the weight values in the chromosome. To save computation, we first set each weight value as 1 and select the top 100 recognition candidates (segmentation-recognition paths) for each training sample.

5. Linguistic Context and Geometric Context

In this chapter, we describe the linguistic context and geometric context, both of them play an important role in the path evaluation criterion for character-position-free on-line handwritten Japanese and Chinese text recognition.

Due to the characters in a handwritten text cannot be segmented unambiguously before recognition, over-segmentation-based method is commonly employed to solve this problem. Therefore, this method may produce many candidate character patterns in the candidate lattice. For each candidate pattern, the character recognizer usually provides not only a unique similar class with the corresponding score, but also top N ($N \geq 1$) candidate classes with scores. The linguistic context can provide valuable information for selecting the optimal class from the top N candidate ones. Moreover, combining the linguistic knowledge, the geometric context and character recognition results, it can verify the candidate patterns, and so improve the text recognition rate.

5.1 Linguistic Context

In handwritten text recognition, the linguistic processing of character recognition results after character segmentation is usually referred to as post processing[16].Due to the character recognizer provides several candidate classes for a candidate character pattern, the selection of the optimal class from the set of candidate classes is based on the linguistic knowledge model. The linguistic knowledge models are usually represented in word dictionaries and statistical language models, such as character-based n-gram [95], and word-based n-gram [32], [96], [97].The word-based n-gram language model is generally based on the syntactic/semantic classes (e.g., parts of speech) of words. Its use in linguistic processing involves the segmentation of text into words, usually by morphological analysis using a lexicon [96], [97].Moreover, the adaptation of writer-specific linguistic dictionaries is beneficial for writer dependent handwritten character recognition [98].Using the linguistic processing, the error rate of off-line handwritten English text is reduced by about 50 percent for single writer data and by about 25 percent for multiple writer data [99].

Recently, the unsupervised language model adaption is proposed for unconstrained off-line handwritten Chinese text patterns, and improves the recognition performance impressively, especially for the ancient domain documents [100].Li et al. [101] applied the recurrent neural network language model (RNNLM), which is superior to the n-gram language models due to its capability to capture long-span history by discriminative leaning using the recurrent neural network, to improve the recognition of off-line handwritten Arabic documents.

Due to the word-based n-gram language models need an additional word segmentation tasks, the simple and effective character-based n-gram model is widely used for handwritten Japanese and Chinese text recognition[8], [9]. Fig.5-1 gives an example to shown character-based and word-based unigram (n = 1), bigram (n = 2) and trigram (n = 3) language model.

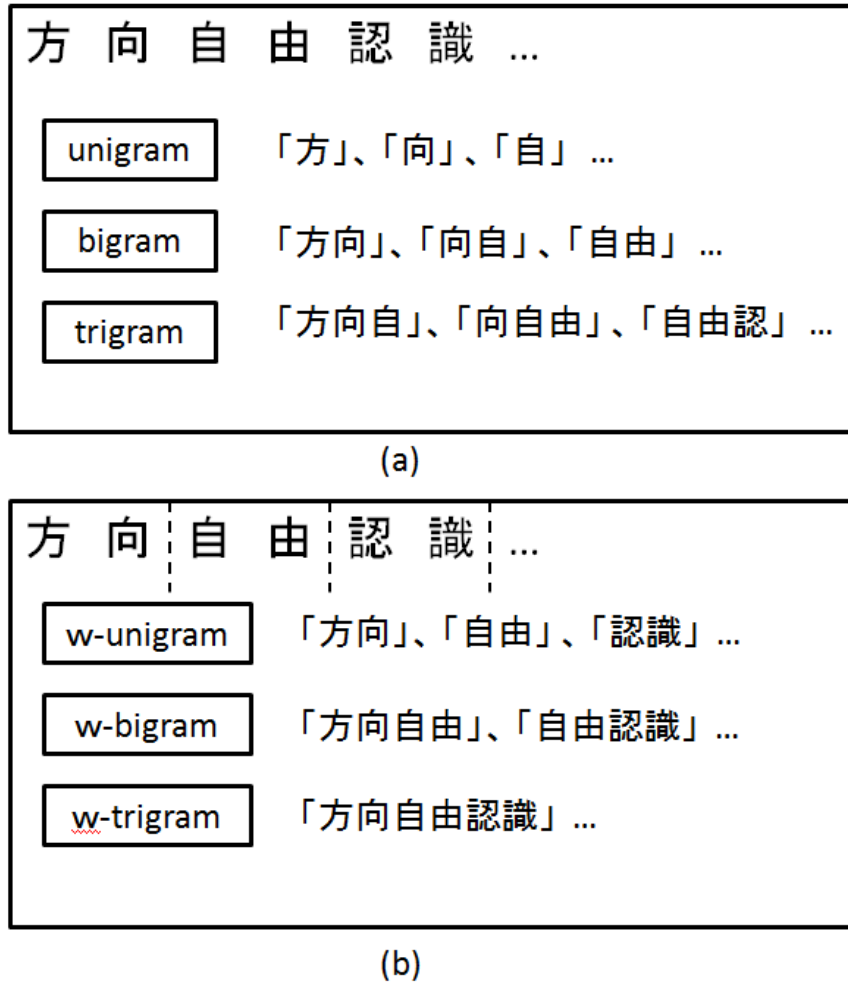


Figure5- 1 Examples of character-based (a) and word-based (b) unigram, bigram and trigram model.

Statistical language modeling involves attempts to capture regularities of natural language in order to improve the performance of various natural language applications, such as machine translations. N-gram models as the most successful statistical language mode have been applied in handwriting recognition, since it can be easily integrated with the character recognition. We will introduce it in the next section.

5.1.1 N-gram language model

The most widely used language models is n-gram language models, where n is called the order of the model. Such model estimates the statistical dependency between n characters or words. Considering the complexity of language models, the order n usually takes 1, 2 or 3, namely unigram, bigram and trigram language model, respectively.

Given a handwritten Japanese/Chinese text or sentence with l characters $W=w_1w_2\cdots w_l$, based on the statistical language model, the priori probability of this sentence $P(C)$ can be decomposed as follows:

$$\begin{aligned} P(W) &= P(w_1)P(w_2/w_1)P(w_3/w_1w_2)\dots P(w_l/w_1\dots w_{l-1}) \\ &= \prod_{i=1}^l p(w_i|w_1w_2 \dots w_{i-1}) \end{aligned} \quad (5-1)$$

Here, we assume that the probability of character w_i being written depends only on the previous characters ($w_1\cdots w_{i-1}$) of the sentence.

In n-gram language models, Eq. (5-1) is transformed into the Eq. (5-2) with changing the probability of character w_i being written depends only on the previous $(n-1)$ characters of the sentence.

$$P(W) = \prod_{i=1}^l p(w_i|w_1w_2 \dots w_{i-1}) = \prod_{i=1}^l P(w_i|w_{i-n+1}^{i-1}) \quad (5-2)$$

where n is called the order of the model. w_{ij} denotes the characters sequence $w_i\cdots w_j$. Even for low orders, the number of equivalence classes becomes quickly intractable. In practice, the unigrams, bigrams and trigrams are commonly used. They are shown in Eq. (5-3), Eq. (5-4), and Eq. (5-5), respectively.

$$P(W) \approx \prod_{i=1}^l p(w_i) \quad (5-3)$$

$$P(W) \approx \prod_{i=1}^l p(w_i|w_{i-1}) \quad (5-4)$$

$$P(W) \approx \prod_{i=1}^l p(w_i|w_{i-2}w_{i-1}) \quad (5-5)$$

The probabilities $P(w_i|w_{i-n+1}^{i-1})$ are estimated from a corpus of training texts using Maximum Likelihood (ML) estimation, namely, by counting the number of times a certain sequence of n .

Characters appears in the corpus of training texts, given by

$$P(w_i|w_{i-n+1}^{i-1}) = \frac{\text{count}(w_{i-n+1}^{i-1})}{\text{count}(w_{i-n+1}^{i-1})} = \frac{C(w_{i-n+1}^{i-1})}{C(w_{i-n+1}^{i-1})} \quad (5-6)$$

where $C(\cdot)$ denotes the number of times the argument is counted from the given training corpus. The model resulting from Eq. (5-6) maximizes the likelihood of the training corpus, which used to obtain the language models. The n-gram statistics language models have several advantages, such as the quick speed due to probabilities of n-gram are stored in pre-computed tables, simple calculation, and generality due to models can be applied to any domain or language, as long as there exists some training corpus.

For the character-position-free on-line handwritten text recognition, we choose the trigram language model which combined not only trigram, but also bigram and unigram models, with considering the computation complexity and effectiveness.

Following the ML estimation, however, the n-gram models face an important problem due to no corpus is large or wide enough to contain all possible n-grams, namely, all the texts of n characters not appearing in the training corpus have zero probability. Moreover, many n-grams appear too few times to allow a good statistical estimation of their probability $P(w_i|w_{i-n+1}^{i-1})$. In order to solve this problem, the smoothing technique is applied. We will introduce it in the next section.

5.1.2 Smoothing algorithm

As many of n-gram probability estimates are going to be zero due to it is impossible that all words are seen in the training text corpus. Whenever a character string W with $P(W) = 0$ during a text recognition task, that is, the character string should not occur, which is too hard discrimination for handwritten text recognition, a recognition error will be made. It helps prevent errors to assign all character strings in non-zero probabilities for handwritten text recognition.

Smoothing technique is used to overcome this problem, i.e. zero probabilities of the unseen n-grams in the given text corpus by redistributing probabilities between seen and unseen events. Smoothing techniques produce more accurate probabilities by adjusting the maximum likelihood estimate of probabilities. Typically, smoothing methods prevent any probability from being zero, but they also attempt to improve the accuracy of the model as a whole. The name smoothing comes from the fact that these techniques tend to make distribution more uniform,

which can be viewed as making them smoother. Especially for the very low probabilities such as zero probabilities are adjusted upward, and high probabilities are adjusted downward.

One simple way of smoothing technique used in practice is the additive smoothing [102], also called Laplace smoothing, which is to pretend each n-gram occurs slightly more often than it actually does for avoiding zero probabilities. Eq. (5-7) is then transformed by following this additive smoothing as follows:

$$P_{\text{add}}(w_i|w_{i-n+1}^{i-1}) = \frac{C(w_{i-n+1}^{i-1}) + \delta}{C(w_{i-n+1}^{i-1}) + \delta|V|} \quad (5-7)$$

where δ is a constant, and subjected to $0 < \delta \leq 1$. V is the vocabulary, or set of all characters considered.

The δ is generally considered as 1, and called add-one smoothing. Let us consider the application of add-one smoothing to bigram probabilities, Eq. (5-8) is simplified as follows:

$$P_{+1}(w_{i-1}|w_i) = \frac{C(w_{i-1}w_i) + 1}{\sum_{\infty} [C(w_{i-1}w_i) + 1]} = \frac{C(w_{i-1}w_i) + 1}{\sum_{\infty} C(w_{i-1}w_i) + |V|} \quad (5-8)$$

where δ is a constant, and subjected to $0 < \delta \leq 1$. V is the vocabulary, or set of all characters considered.

The δ is generally considered as 1, and called add-one smoothing. Let us consider the application of add-one smoothing to bigram probabilities, Eq. (5-9) is simplified as follows:

$$P_{+1}(w_i|w_{i-1}) = \frac{C(w_{i-1}w_i) + 1}{C(w_{i-1}) + |V|} \quad (5-9)$$

Many other smoothing techniques have been introduced in the literature [102]. Such as the simple interpolation, Katz smoothing, back off Kneser-Ney smoothing and Interpolated Kneser-Ney smoothing. We will describe the simple interpolation algorithm.

The simple interpolation is a combine technique in language modeling to simply interpolate them together. For instance, if one has a trigram model, a bigram model, and a unigram model, then

$$P_{\text{Interpolation}}(w_i|w_{i-2}w_{i-1}) = \lambda_1 P(w_i|w_{i-2}w_{i-1}) + \lambda_2 P(w_i|w_{i-1}) + \lambda_3 P(w_i) \quad (5-10)$$

where λ_1 , λ_2 and λ_3 are parameters with constraint that $0 \leq \lambda_1, \lambda_2$ and $\lambda_3 \leq 1$. In practice, to ensure no word is assigned zero probability, we commonly interpolate with the uniform distribution $P(w_i) = 1/\text{size of vocabulary}$, we also need to deal with the case when, for instance, the trigram context $c_{i-2}c_{i-1}c_i$ has never been seen, namely $C(w_{i-2}w_{i-1}w_i) = 0$. In this case, we use an interpolated bigram model, etc. Given its simplicity, simple interpolation works surprisingly well, but other techniques, such as Katz smoothing, work even better, but need much more training corpus.

In our study, we use this smooth method for trigram language model, which combines the unigram, bigram and trigram with parameters, where the parameters subject to $\lambda_1 + \lambda_2 + \lambda_3 = 1$. It is reduced to unigram or bigram when w_i is the first or second character of a sentence. Moreover, to reduce the model size, we set empirically a threshold to prune the low trigrams probabilities.

5.2 Geometric Context

In handwritten Japanese/Chinese text recognition, over-segmentation-based method is commonly employed to overcome the character segmentation problem, due to it is infeasible to segment character reliably prior to recognition. The geometric context, which includes the compatibility of character size, position and between-character relationship with respect to the text layout, can help disambiguate the uncertainty in character segmentation. Especially in Japanese language, the position of the small Kana such as “tsu” is obviously different with the normal Kanji. Furthermore, the recognition accuracy will be improved by incorporating the geometric context with character recognition and linguistic context in the candidate segmentation and recognition path evaluation.

In on-line handwritten Japanese text recognition, Nakagawa et al. [10] incorporated the likelihood of geometric features into the path scores, but only simple features are used, such as character size, inter-character and between-character gap. Zhou et al. [37] incorporated the geometric context with character recognition and linguistic context into a united framework to overcome the effect of string length variability and improve the recognition performance. Here, the geometric context models are made by a statistical method, including class-dependent unary and binary geometric features with more features. Recently, Zhu et al. [8] combined the more geometric context with the character recognition, linguistic context and character segmentation to improve recognition accuracy. The geometric features include the size feature, character inner-gap feature, and class-dependent unary and binary position features.

In on-line handwritten Chinese text recognition, the employed geometric context including more features (class-dependent unary and binary geometric features, and class-independent

unary and binary geometric features)[46], [9]. Compared to [8], the main difference is that it uses class-independent unary and binary geometric features, using simple SVM model. Yin et al. [103] integrated the geometric features to improve the performance of text alignment in Chinese annotation system. Wu et al. [104] proposed an improved binary geometric model that combines single-character and between-character features to improve significantly the numeral string recognition performance on the NIST special database [105].

We will introduce the geometric features used in the character-position-free handwritten text recognition system, including the character size feature, character inner-gap feature, and unary position feature of a candidate character pattern, and binary position feature between candidate character patterns.

The character size feature (or shape feature), namely the term b_i in Eq. (5-7), is composed of the height and width of the bounding box of a candidate character pattern. Fig.5-2 shows an example of size features of candidate character patterns.

The character inner-gap feature of a candidate pattern, namely the term q_i in Eq. (5-7), is obtained by projecting the candidate character pattern into the vertical and horizontal directions, splitting each of their histograms into 3 slices, finding a gap or gaps in each slice, and summing total lengths of gaps. Hence, the inner-gap feature vector includes 6 values. Fig.5-2 shows an example of the inner-gap feature of a character pattern.

The class-dependent unary position feature, namely the term p_i^u in Eq. (5-7), consists of two vertical distances from the horizontal center of a text line to the top and bottom of the bounding box of a candidate character pattern, as shown in Fig. 5-2.

The class-dependent binary position feature, namely the term p_i^b in Eq. (5-7), is composed of a vertical distance between the top edges of the bounding boxes of two adjacent candidate character patterns in a text line and that between the bottom edges of the bounding boxes, as shown in Fig.5-3.

Due to Japanese and Chinese language are large characters set including thousands of character classes, it is almost impossible to get sufficient training samples covering every class pair. A feasible method is that using cluster method to reduce the number of classes. The character classes are then clustered into six super-classes by grouping the mean vectors of the unary geometric features of all character classes on a training text set using the k-means algorithm. Hence, a pair of successive characters belongs to one of 36 binary super-classes. The training text character samples, re-labeled to six unary super-classes, are used to estimate the Gaussian probability density functions of 36 binary super-classes. Then, the binary geometric

probability $P(p_i^b | c_{i-1}, c_i)$ is substituted by $P(p_i^b | \tilde{c}_{i-1}, \tilde{c}_i)$, where \tilde{c}_{i-1} and \tilde{c}_i are the unary super-classes of c_{i-1} and c_i .

We normalize the above 4 features (b_i , q_i , p_i^u , and p_i^b) by the *acs* (average character size). Then, we assume $P(b_i | c_i)$, $P(q_i | c_i)$, $P(p_i^u | c_i)$, and $P(p_i^b | \tilde{c}_{i-1}, \tilde{c}_i)$ to be normal distributions and model their logarithms by a quadratic discriminant function (QDF).

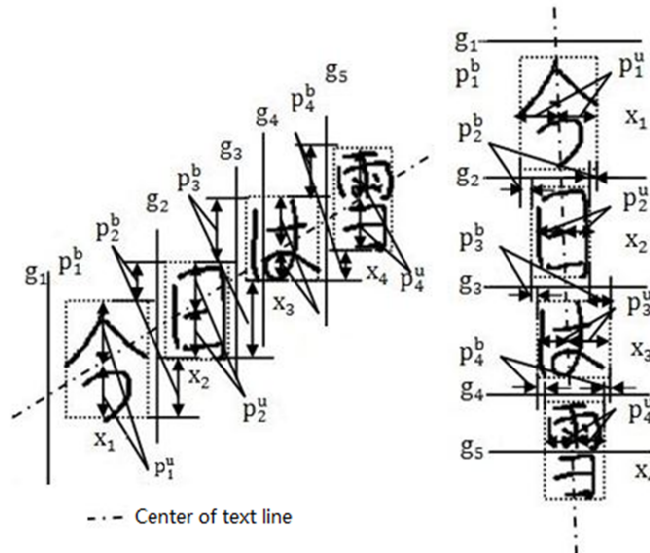


Figure5- 2 Geometric features.

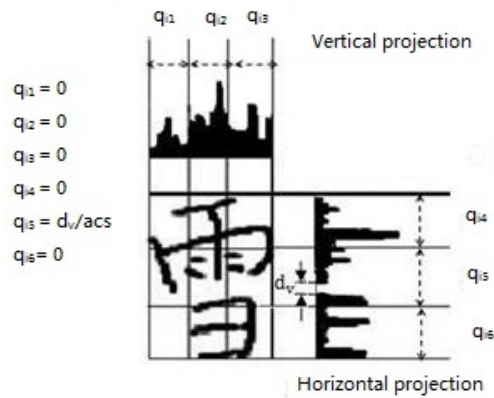


Figure5- 3 Inner gap feature within a character pattern

6. Experiments

In this chapter, we describe several experiments on character-orientation-free and line-direction-free on-line handwritten Japanese text datasets.

- ① Evaluation of Dimensionality Reduction
- ② Evaluation of Parameter Optimization by GA
- ③ Comparison with the Segmentation-updated System
- ④ Performance on Arbitrary Line Direction and Character Orientation
- ⑤ Performance on Artificially Rotated Text lines
- ⑥ Performance on Time Cost and Memory Requirement for Dictionaries

6.1 Kondate Database

Kondate [69] is a database of on-line handwritten patterns mixed of texts, figures, tables, maps, diagrams and so on. In this research, we only use the part of on-line handwritten texts, which initially has been collected in Japanese from 100 people at Tokyo University of Agriculture and Technology (TUAT) in Japan.

As for on-line handwritten Japanese texts in Kondate, the most text patterns were collected by writing natural sentences taken from a Japanese newspaper on display integrated tablets. The writing style was not constrained so that most of the characters were written fluently although some people write in regular style due to their writing habit. Moreover, the writers write freely without any writing grids and even without guidelines.

Therefore, Kondate database covers any direction text patterns, such as horizontal, vertical, diagonal, horizontal and vertical mixed text and so on. As shown in Table6-1. From Table 6-1, we can see that two categories: “copy writing” to cover the categories and “free writing” to collect casually and naturally written patterns as described below.

Copy writing: In order to the categories, we present a participant with sample patterns, Then, a participant writes the identical contents. Since, we do not specify the writing order, the writing order may be different although the result looks similar. For the copy writing, we prepared our own sample patterns based on our experiences. Although, we employed sample sentences from a newspaper for fairness in Kuchibue and Nakayosi[105]. The copy right issue with the newspaper caused a problem for distribution.

Free writing: In order to get natural patterns, we ask a participant to write about a topic. For instance, we ask “Please write (draw) a map of your way from your home to your school or office”, “Solve the following equation” and so on,

The copy writing is effective to cover the categories and easy to get the ground truth but has the problem to collect real patterns. On the other hand, the free writing affords participants to write or draw real patterns, but coverage is not assured and ground-truthing is not straightforward.

Table6- 1 Pages in *HANDS-Kondate_t_bf-2001-11* (100 people).

Page	Text	Line direction	Character orientation	Content
1-11	copy	right	down	horizontal writing
12-22	copy	down	down	vertical writing
23	free	vertical, horizontal and slanting	unspecified	route
24	free	vertical, horizontal and slanting	unspecified	Comments about news
25	free	vertical, horizontal and slanting	unspecified	Comments about the collection
26	copy	mixture of vertical and horizontal	down	Information about menu
27	copy	mixture of horizontal and slanting	slanting	Information about product
28	copy	mixture of vertical and horizontal	down, up, left and right	Lay out of “mah-jong”

6.2 Datasets

We employ a Japanese on-line handwriting database Nakayosi [19], which stores 1,695,689 patterns for 4,438 categories of Kanji of Chinese origin, two sets of Kana (Hiragana and Katakana) of phonetic characters, English upper case and lower case letters, numerals, and so on to train the character recognizer and geometric scoring functions. The character recognizer combines off-line and on-line recognition methods by normalizing the recognition scores to conditional probabilities $P(x_i/C_i)$ [20]. Four QDF classifiers are trained for the geometric scores $P(b_i/C_i)$, $P(q_i/C_i)$, $P(p_i^u/C_i)$ and $P(p_i^b/C_{i-1}, C_i)$.

We prepare the tri-gram table for the linguistic context from the year 1993 volume of the ASAHI newspaper and the year 2002 volume of the NIKKEI newspaper. The data size of the tri-gram is reduced to 10MB by suppressing non-occurring terms, neglecting a small number of occurrences, and quantizing the logarithm values of tri-gram probabilities. In order to train text recognition parameter and test performance, we employ the database of character-orientation

and line-direction free handwritten on-line text *HANDS-Kondate_t_bf-2001-11* collected from 100 people, as shown in Table 6-1. Pages 1 through 11 are just horizontal handwritings and pages 12 through 22 are just vertical lines. Pages 23 through 28 are mixture of horizontal, vertical and slanting text lines with various character orientations. Especially, pages 24 and 25 have freely handwritten patterns under certain topics as shown in Fig. 6-1.

We prepare sample patterns by separating handwritten text in all the pages in the database into text line elements, normalizing their character orientations and classifying them into 4 line directions (*R*, *L*, *U* and *D*).

For each line direction, we employ 4-fold cross validation where we use 75 persons' text for training and the remaining 25 persons' text for testing. We select one group among the 4 groups as the testing set *i* (*i* =1 to 4) and merge all the remaining groups (75 persons' patterns) as the training set *i*. We employ the training set *i* to train the SVM classifier for over-segmentation and a set of weighting parameters, because 21 over-segmentation features and geometric features for path evaluation are different for each line direction. Then, we evaluate the performance for the testing set *i*. We turn the role of training and testing sets and take the average of the 4 turns. Table 6-3, 6-4, 6-5, 6-6 are the data sets.

The statistics of the training sample (Train.) and testing samples (Test.) for the extracted text line elements are listed in Table 6-2, where *N_{sp}* is the number of true segmentation points, *N_{nsp}* is that of true non-segmentation points, *N_{ac}* is the average number of characters in a text line element and *N_{al}* is the average number of characters written by one person, respectively.

Page23	Page24	Page25
Page26	Page27	Page28

Figure6- 1 Examples in HANDS-Kondate_t_bf-2001-11 database from Page23 to Page28.

Table6- 2 Number of samples in training and testing sets for each direction

Method Line direction	R		L		D		U	
	Train	Test	Train	Test	Train	Test	Train	Test
Text lines	42423	14141	498	166	26604	8868	267	89
English letters	17973	5991	18	6	12672	4224	12	4
Numeral	68295	22765	177	59	31053	10351	15	5
Kana	163287	54429	2253	751	126609	42203	921	307
Kanji	151443	50481	1335	445	106230	35410	537	179
Others	46824	15608	231	77	29682	9894	99	33
Nsp	405399	207894	3516	1172	279642	93214	1317	439
Nnsp	1223712	335143	11859	3953	860442	286814	4485	1498
Nac(average)	10.6	10.6	8.1	8.0	11.5	11.5	5.9	5.8
Nal(average)	1492.7	1492.7	13.4	13.4	1020.8	1020.8	5.3	5.3

Table6- 3 Number of samples in training and testing sets for each direction (dataset1)

Method Line direction	Data type	Text lines	Nsp	Nnsp
R	Train	10552	50737	98775
	Test	3635	16721	33509
L	Train	123	14	653
	Test	35	2	223
D	Train	6811	20973	68409
	Test	2031	6754	22582
U	Train	71	23	301
	Test	16	4	46

Table6- 4 Number of samples in training and testing sets for each direction (dataset2)

Method Line direction	Data type	Text lines	Nsp	Nnsp
R	Train	11068	51822	101134
	Test	3119	15636	31150
L	Train	103	8	530
	Test	55	8	346
D	Train	6478	20355	67957
	Test	2364	7372	23034
U	Train	57	17	217
	Test	30	10	130
	Test	3016	4	46

Table6- 5 Number of samples in training and testing sets for each direction (dataset3)

Method Line direction	Data type	Text lines	Nsp	Nnsp
R	Train	10494	49798	98507
	Test	3693	17660	33777
L	Train	123	11	722
	Test	30	5	154
D	Train	6526	20620	67461
	Test	2316	7107	23530
U	Train	55	16	203
	Test	32	11	144
	Test	16	4	46

Table6- 6 Number of samples in training and testing sets for each direction (dataset4)

Method Line direction	Data type	Text lines	Nsp	Nnsp
<i>R</i>	Train	10447	50017	98436
	Test	3740	17441	33848
<i>L</i>	Train	120	15	723
	Test	38	1	153
<i>D</i>	Train	6711	21233	69146
	Test	2131	6494	21845
<i>U</i>	Train	78	25	320
	Test	9	2	27
	Test	16	4	46

The following subsections report the effects of dimensionality reduction, parameter optimization by GA, comparison with the segmentation-updated system, performance on artificially rotated text lines in comparison with the Onuma et al. system [11] and the time complexity and memory requirement for dictionaries of the system.

The performance is evaluated by the f -measure for segmentation and the character recognition rate Cr as shown in Eq. (6-1) where r is recall and p is precision respectively. The recall rate measures the tolerance to search errors, while the precision rate measures the tolerance to search noises.

The experiments are implemented on Intel(R) Core(TM) i7-3770S 3.10GHZ with 4.0GB memory.

$$f = \frac{2}{1/r + 1/p}$$

$$r = \frac{\text{number of correctly detected segmentation points}}{\text{number of true segmentation points}} \quad (6-1)$$

$$p = \frac{\text{number of correctly detected segmentation points}}{\text{number of detected segmentation points (including false)}}$$

6.3 Evaluation of Dimensionality Reduction

In order to investigate the effect of the feature reduction using PCA to reduce the extracted 21 geometric features for over-segmentation, we test the results of reducing the 21 geometric features. Table 6-7 presents the average performance on the testing sets, where the average

character recognition time T and the memory size of the system are shown as well as the f-measure for segmentation and the character recognition rate Cr .

Table6- 7 Performance on reduced dimensionalities.

Dimension Line direction	Reduced dimensionality											
	21	19	17	15	13	11	9	7	5	3	1	
Memory (MB)	12.88	11.74	10.79	9.78	8.54	6.98	5.75	4.73	3.81	2.96	1.52	
R	f	0.986	0.984	0.983	0.983	0.983	0.983	0.984	0.984	0.984	0.983	0.983
	$Cr(\%)$	92.22	92.15	91.87	91.87	91.86	91.87	91.89	91.90	91.89	91.75	91.83
	$T(\text{ms})$	54.0	53.5	49.3	49.1	48.9	48.9	47.6	47.4	46.1	44.3	39.8
L	f	0.980	0.980	0.980	0.980	0.980	0.980	0.980	0.980	0.980	0.972	0.972
	$Cr(\%)$	92.93	92.93	92.93	92.93	92.93	92.93	92.93	92.93	92.93	90.96	90.96
	$T(\text{ms})$	37.8	37.8	37.8	37.8	37.8	37.8	37.8	37.8	37.8	23.5	23.5
U	f	0.962	0.962	0.962	0.962	0.962	0.962	0.962	0.962	0.962	0.962	0.962
	$Cr(\%)$	91.52	91.52	91.52	91.52	91.52	91.52	91.52	91.52	91.52	91.52	91.52
	$T(\text{ms})$	79.9	79.9	79.9	79.9	79.9	79.9	79.9	79.9	79.9	79.9	58.2
D	f	0.991	0.986	0.986	0.988	0.988	0.988	0.988	0.989	0.988	0.989	0.986
	$Cr(\%)$	91.60	91.07	91.07	91.09	91.09	91.10	91.14	91.16	91.10	91.13	91.02
	$T(\text{ms})$	49.6	49.3	49.3	48.8	48.6	48.6	47.9	47.7	47.3	47.0	45.9

From the results, we can see that the 21 dimensionality of the original features achieves the best recognition and segmentation accuracy while it consumes the largest memory space and recognition time although it is not so serious for a common personal PC. Other reduced dimensionalities such as 5 can reduce the memory size and recognition time largely without losing recognition and segmentation accuracy significantly. In order to keep the high recognition rate in the following experiments, we use the 21-dimensional features.

6.4 Evaluation of Parameter Optimization by GA

In order to justify weighting parameter optimization by GA, we draw a comparison between GA and the minimum classification error (MCE) criterion [21] optimized by stochastic gradient decent [22] (MCE-SGD). MCE-SGD is to find the optimal parameter vector λ by minimizing the following difference between the scores of the most confusing text class and that of the correct one:

$$L_{MCE}(\lambda, X) = \sigma(\max(\text{Score}_{\text{Incorrect}}) - \text{Score}_{\text{correct}})$$

$$\sigma(x) = (1 + e^{-x})^{-1}$$

(6-2)

Score_{correct} = Score of the correct path in a sr – lattice

Score_{Incorrect} = Scores of the incorrect paths in a sr – lattice

Table6- 8 Performance optimized by GA and MCE-SGD v.s segmentation-updated system.

Method Line direction	GA		MCE-SGD		Segmentation-updated System[12]	
	<i>f</i>	<i>Cr</i> (%)	<i>f</i>	<i>Cr</i> (%)	<i>f</i>	<i>Cr</i> (%)
R	0.9864	92.22	0.9855	92.07	0.9660	73.61
L	0.9809	92.93	0.9748	91.95	0.9838	80.75
U	0.9624	91.52	0.9703	91.10	0.9897	80.23
D	0.9710	91.60	0.9874	90.89	0.9647	75.83

Table 6-8 shows the average performance on the testing sets, from which we can see that the optimization GA produces better performance than MCE-SGD.

6.5 Comparison with the Segmentation-updated System

We also compare the performance of our proposed system and that by the segmentation-updated method in [12], which can be tested since character orientation is already normalized. It used an one-stage classification scheme on over-segmentation and applied a recognition model that could be viewed as a special case of the model employed in this paper by setting $\lambda h_1 = 1, \lambda h_2 = 0$ ($h = 1, \dots, 7$) without using the terms related to k_i (number of primitive segments composing a character parameter) in Eq.(6-3).

$$f(\mathbf{X}, \mathbf{C}) = \sum_{i=1}^n \left(\begin{aligned} & (\lambda_{41} + \lambda_{42}(k_i - 1)) \log P(C_i | C_{i-1}, C_{i-2}) + \\ & (\lambda_{21} + \lambda_{22}(k_i - 1)) \log P(b_i | C_i) + (\lambda_{51} + \lambda_{52}(k_i - 1)) \log P(q_i | C_i) + \\ & (\lambda_{41} + \lambda_{42}(k_i - 1)) \log P(x_i | C_i) + (\lambda_{51} + \lambda_{52}(k_i - 1)) \log P(p_i^u | C_i) + \\ & (\lambda_{61} + \lambda_{62}(k_i - 1)) \log P(p_i^b | C_i, C_{i-1}) + \\ & \lambda_{71} \log P(g_{ji} | Sb) + \lambda_{72} \sum_{j=j_i+1}^{j_i+k_i-1} \log P(g_j | Sw) \end{aligned} \right) + \lambda n \quad (6-3)$$

We add the performance by the segmentation-updated system in Table 6-9 and we can see that the character recognition rate has been largely improved.

6.6 Performance on Arbitrary Line Direction and Character Orientation

We test the performance on the 6 pages (23~28). Table 6-9 shows the average performance on the testing sets in comparison with the Onuma et al. system, where scores by the Onuma et al. system are quoted from [11]. We can see the large improvement from the Onuma et al. system due to the update of line-segmentation, over-segmentation and path evaluation, although we do not validate the Onuma et al. system by cross validation.

Table6- 9 Performance on mixture of vertical, horizontal and slanting lines.

Method Page No	Proposed System		Onuma et al. System[11]	
	<i>f</i>	<i>Cr</i> (%)	<i>f</i>	<i>Cr</i> (%)
23	0.9878	92.33	0.7523	63.87
24	0.9792	90.70	0.7368	62.66
25	0.9767	89.62	0.7118	60.51
26	0.9763	89.43	0.8698	72.81
27	0.9675	86.45	0.7245	62.10
28	0.9933	90.58	0.7538	64.61

P23 (topic about route)	The result of recognition
	<p>家を出て、JR八王子駅まで自転車でドビューンと円分。中央線でガタゴト52分の東入、金井駅で下車○それから徒歩でサ泰和を左にまがって薬局を右にまかいつてまつすぐ、つきあー川を左へまがると農工大に到着0 処理時間：10233[ms]</p>
P26(topic about menu)	The result of recognition

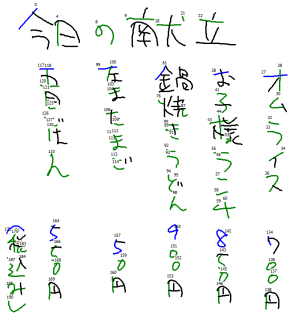
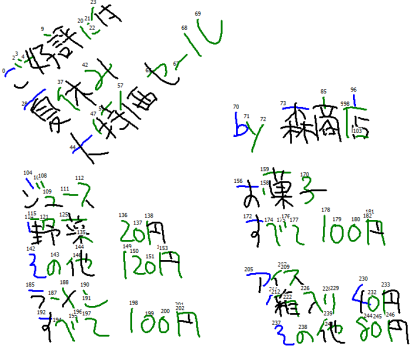
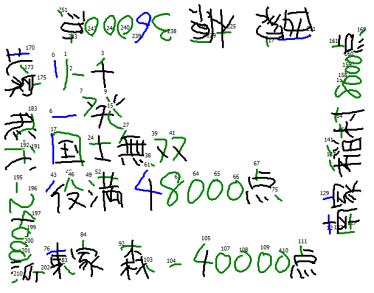
	<p>今日の献立オムライスお子様ランチ鍋 焼きうどんーはまたまで五日ごはん 700円850円900円50円550円~税込みし 処理時間：2902[ms]</p>
<p>P27(topic about product)</p>	<p>The result of recognition</p>
	<p>ご好評に斥剣み大特価セール▽、1森 商店ジュース野菜20円その他120円お 菓子すべて100円ラーメンすべて100円 アイス箱入り40円その他80円 処理時間：3909[ms]</p>
<p>P28(topic about product with “mah-jong” layout)</p>	<p>The result of recognition</p>
	<p>リーチー伴し国土無双役満48000点東 家森ー40000点南家稻村8000点北家飛 代ー24000点西家木原36000点 処理時間：3863[ms]</p>

Figure6- 2 Page of 23-28 and the result of recontion

P23-P25 the recognition errors due to line segmentation and character recognition. To solve this problem, we should to improve the character recognition accuracy. Increasing the number of candidate classes can reduce the missing of correct class.

P26 the recognition errors due to the recognition of parentheses.To solve this problem, we should to add the parenthese into the dictionary.

P27 the recognition errors due to the recognition of the text lines of characters(ご好評に付夏休み 大特価セール), the mis-normalization of the line direction and character orientation. To solve this problem, we should improve the estimation of line direction and character orientation.

P28 the recognition errors due to the recognition of a minus sign.To solve this problem, should to add the a minus sign into the dictionary.

6.7 Performance on Artificially Rotated Text lines

Since the amount of text lines in the above 6 pages (23~28) is not many, we prepare artificially rotated text lines by rotating the original 22 pages (horizontal handwritings in pages 1 through 11 and vertical lines in pages 12 through 22 by the amount of 30, 90, 130 and 240 degrees as shown in Fig. 6-3.

30degree	90degree
130degree	240degree

Figure6- 3 Examples of rotated handwritten text lines.

Table 6-10 shows the average performance on the testing sets in comparison with the Onuma et al. system, where scores by the Onuma et al. system are quoted from [11]. We can see that the proposed method works well without any degradation for those artificially rotated text

lines and its performance is far better than the Onuma et al. system due to the enhancement in line-segmentation, over-segmentation and path evaluation, although we do not validate the Onuma et al. system by cross validation.

Table6- 10 Performance on rotated horizontal/vertical handwritings.

Method Degree	Proposed System		Onuma et al. System[11]	
	<i>f</i>	<i>Cr(%)</i>	<i>f</i>	<i>Cr(%)</i>
Original	0.9899	92.92	0.8386	71.52
30degree	0.9865	91.78	0.8212	70.25
90degree	0.9896	92.44	0.8375	71.57
130degree	0.9884	92.74	0.8232	69.86
240degree	0.9898	92.76	0.8229	70.16

6.8 Evaluation of Time Cost and Memory Requirement for Dictionaries

Finally, we present the time and space complexity of the system as shown in Table 6-11, 6-12. The time and memory requirement of the Onuma et al. system was 16.84 m sec, 7.87MB, respectively.

The time complexity has been doubled and the Memory requirement has been tripled. Nevertheless, the recognition speed is quick enough for practical applications and the memory requirement is practical, which is almost the same as the recognizer for horizontal.

Table6- 11 Evaluation of time cost.

Process	Processing time/character(ms)
Segmentation of handwriting into text line elements	0.61
Estimation and normalization of character orientation	
Quantization of line direction	
Over-segmentation	3.22
Construction of sr-lattice	0.10
Search and recognition	27.32
Total recognition time	31.25

Table6- 12 **Memory requirement for dictionaries.**

Dictionaries		Memory Size
Tri-gram		10.67MB
Geometric features	Size $P(b_i C_i)$ and inner gap $P(q_i C_i)$	542KB
	Unary $P(p_i^u C_i)$	270KB
	Binary $P(p_i^b C_{i-1},C_i)$	280KB
Character recognizer		15.25MB
4 SVM classifiers for over-segmentation		2.39MB
Total memory requirement size		29.37MB

7. Conclusion and Future Work

In this chapter, we draw the conclusion of this thesis, and give several directions for the future works.

7.1 Conclusion

This paper has presented significant improvements in our line-direction-free and character-orientation-free on-line handwritten Japanese text recognition system. Through updating the text line segmentation and over-segmentation and then integrating them into a robust context integration model, the system achieves high recognition rates on on-line handwritten text of arbitrary character orientation and line direction comparable with horizontal text with normal character orientation.

The results of experiments on text from the *HANDS-Kondate_t_bf-2001-11* database demonstrate significant improvements in the character recognition rate compared with the previous systems. Rightward 92.22%, leftward 92.93%, upward 91.52% and the downward is 91.60% respectively. The recognition rate mixture of horizontal, vertical and slanting lines of text with arbitrary character orientation is about 90%.

The time and memory requirement of the Onuma et al. system [11] was 16.84 m sec, 7.87MB, respectively (29.37MB). The time complexity has been doubled (31.25ms) and the Memory requirement has been tripled. Nevertheless, the recognition speed is quick enough for practical applications and the memory requirement is practical, which is almost the same as the recognizer for horizontal.

There remain several works to improve the performance. The gravity center information [13] might be added for line direction estimation. Module organizations must be refined to incorporate the system for ink search and other applications.

7.2 Future work

Fig. 7-1 shows some examples of misrecognition and mis-segmentation given by the proposed model. For each example, the upper line is the written text, and the lower line is the recognition result followed by the correct result where the recognition errors are highlighted by underlines. We observed three major sources causing segmentation-recognition errors.

- し折衣で"前手の切躰
- (a) 超切の手前で右折し、(踏切の手前で右折し、)
- 例文が
- (b) 例文が○ (例文が。)
- 発言など...
- (c) 発言など...(発言など...)
- 調布市染地2-3-49
- (d) 調布市染地2. 33. 49(調布市染地2—33—49)
- そこから
井ノ頭
- (e) そこ架、頭線 (そこから井ノ頭)
- 人気である
大
- (f) を気である (大人気である)

The text below each string pattern is the recognition result, followed by correct result

Figure7-1 Examples of recognition errors.

(1) Problem of character recognition: Figure7-1 (a)-(b) show recognition errors due to character recognition, where the correct answers are not within the top 10 candidate classes output by the character recognizer for each character pattern. To solve this, we need to improve the character recognition accuracy. Increasing the number of candidate classes can reduce the missing of correct class, but slows down the recognition speed.

(2) Problem of path evaluation: Figure7-1 (c)-(d) show recognition errors due to path evaluation. Correct character answers are within the top 10 candidate classes but the path evaluation fails to find the correct one. To solve this, we should improve the scores of linguistic context and geometric features.

(3) Problem of over-segmentation: Figure7-1 (e)-(f) show recognition errors due to over-segmentation. There are mis-segmentations at the over-segmentation stage. To solve this, we need to improve the accuracy of over-segmentation.

For interface research, it is necessary to recognize character strings freely written in directions by electronic boards and tablets, and to be able to edit with pen and finger. It should be able to change the size and the font of the characters, and the line direction with gestures.

For application research, we are considering handwriting freely written on electronic boards and tablets, from which we can search from the Web or get the machine translate and save the results. If you learn and use handwritten notes in class, you can use it and save it, you also can review it at the next class and proceed to the next lesson.

Reference

- [1] S. Inatani, T. Van Phan, M. Nakagawa, "Comparison of MRF and CRF for Text/Non-text classification in Japanese Ink Documents," Proc. 14th ICFHR, pp. 684-689, 2014.
- [2] T. Van Phan, M. Nakagawa, "Text/Non-Text Classification in Online Handwritten Documents with Recurrent Neural Networks," Proc. 14th ICFHR, pp. 23-28, 2014.
- [3] H. Murase, T. Wakahara and M. Umeda, "Online Writing-Box Free Character String Recognition by Candidate Character Lattice Method," (in Japanese) Trans. of IEICE Japan, vol. J68-D, no. 4, pp. 765-772, 1985.
- [4] M. Okamoto, H. Yamamoto, T. Yoshikawa and H. Horii, "Online Character Segmentation Method by Means of Physical Features," (in Japanese) Technical Report of IEICE Japan, vol. 95, no. 43, pp. 93-99, 1995.
- [5] H. Aizawa, T. Wakahara and K. Odaka, "Real-Time Handwritten Character String Segmentation Using Multiple Stroke Features," (in Japanese) Trans. of IEICE Japan, vol. J80-D-II, no.5, pp. 1178-1185, 1997.
- [6] T. Fukushima and M. Nakagawa, "On-line Writing-box-free Recognition of Handwritten Japanese Text Considering Character Size Variations, Proc. 15th ICPR, vol. 2, pp. 359-363, 2000.
- [7] S. Senda and K. Yamada, "A Maximum-Likelihood Approach to Segmentation-Based Recognition of Unconstrained Handwriting Text," Proc. 6th ICDAR, pp. 184-188, 2001.
- [8] B. Zhu, X.-D. Zhou, C.-L. Liu and M. Nakagawa, "A Robust Model for On-line Handwritten Japanese Text Recognition," Int. J. Document Anal Recognition, vol. 13, no. 2, pp. 121-131, 2010.
- [9] X.-D.Zhou, D.-H. Wang, F. Tian, C.-L. Liu and M. Nakagawa, "Chinese/Japanese text recognition using semi-Markov conditional random fields," IEEE Trans. on PAMI, vol. 35, no. 10, pp. 2484-2497, 2013.
- [10] M. Nakagawa, B. Zhu and M. Onuma, "A Model of On-line Handwritten Japanese Text Recognition Free from Line Direction and Writing Format Constraints," IEICE Trans. Inf. & Syst., vol. E88-D, no. 8, pp. 1815-1822, 2005.
- [11] M. Onuma, A. Kitadai, B. Zhu and M. Nakagawa, "An On-line Handwritten Japanese Text Recognition System Free from Line Direction and Character Orientation Constraints," IEICE Trans. Inf. & Syst., vol. E88-D, no. 8, pp. 1823-1830, 2005.

- [12] B. Zhu and M. Nakagawa, "Segmentation of on-line freely written Japanese text using SVM for improving text recognition," *IEICE Trans. Inf. & Sys.*, vol. E91-D, no. 1, pp.105-113, 2008.
- [13] T. Long and L.-W Jin, "A novel orientation free method for online unconstrained cursive handwritten Chinese word recognition," *Proc.19th ICPR*, pp.1-4, 2008.
- [14] B. Zhu and M. Nakagawa, "Online Handwritten Chinese/Japanese Character Recognition," Xiaoqing Ding ed. *Advance in Character Recognition*, ISBN 978-953-51-0823-8, InTech, Chapter 3, pp. 51-68 2012.
- [15] R. Plamondon and S.N. Srihari, "On-line and off-line handwriting recognition: a comprehensive survey," *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 22, no. 1, pp. 63-84, 2000.
- [16] C.C. Tappert, C.Y. Suen and T. Wakahara, "The state of the art in on-line handwriting recognition," *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 12, no. 8, pp. 787-808, 1990.
- [17] M. Nakagawa, "Non-keyboard input of Japanese text on-line recognition of handwritten characters as the most hopeful approach," *Journal of Information Processing*, vol. 13, no. 1, pp. 15-34, 1990.
- [18] C.L. Liu, S. Jaeger and M. Nakagawa, "Online recognition of Chinese characters: the state-of-the-art," *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 26, no. 2, pp. 198-213, 2004.
- [19] B. Zhu and M. Nakagawa, "Recent trends in online handwritten character recognition," (in Japanese) *IEICE Trans. Inf. & Sys.*, vol. J95-D, no. 4, pp. 335-340, 2012.
- [20] L.Y. Tseng and R.C. Chen, "Segmenting handwritten Chinese characters based on heuristic merging of stroke bounding boxes and dynamic programming," *Pattern Recognition Letters*, vol. 19, no. 10, pp. 963-973, 1998.
- [21] Y. Lu, C.L. Tan, P.F. Shi, and K.H. Zhang, "Segmentation of handwritten Chinese characters from destination addresses of mail pieces," *International Journal of Pattern Recognition and Artificial Intelligence*, vol. 16, no. 1, pp. 85-96, 2002.
- [22] S. Zhao, Z. Chi, P. Shi and H. Yan, "Two-stage segmentation of unconstrained handwritten Chinese characters," *Pattern Recognition*, vol. 36, no. 1, pp. 145-156, 2003.
- [23] X. Wei, S. Ma, and Y. Jin, "Segmentation of connected Chinese characters based on genetic algorithm," *Proceedings of the 8th International Conference on Document Analysis and*

Recognition, Seoul, Korea, pp. 645-649, 2005.

- [24] Z. Liang and P. Shi, "A metasynthetic approach for segmenting handwritten Chinese character strings," *Pattern Recognition Letters*, vol. 26, no. 10, pp. 1498-1511, 2005.
- [25] N. Furukawa, J. Tokuno and H. Ikeda, "Online character segmentation method for unconstrained handwriting strings using off-stroke features", *Proceedings of the 10th International Workshop on Frontiers in Handwriting Recognition(IWFHR)*, La Baule, France, 2006.
- [26] C.L. Liu, H. Sako and H. Fujisawa, "Effects of classifier structures and training regimes on integrated segmentation and recognition of handwritten numeral strings," *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 26, no. 11, pp. 1395-1407, 2004.
- [27] M. Cheriet, N. Kharma, C.L. Liu and C.Y. Suen, "Character recognition systems - A guide for students and practitioners," *John Wiley & Sons, Inc., Hoboken, New Jersey*, 2007.
- [28] M. Mohamed and P. Gader, "Handwritten word recognition using segmentation-free hidden Markov modeling and segmentation-based dynamic programming techniques," *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 18, no. 5, pp. 548-554, 1996.
- [29] H. Murase, T. Wakahara and M. Umeda, "Online writing-box free character string recognition by candidate character lattice method," (in Japanese) *IEICE Trans. Inf. & Sys.*, vol. J68-D, no. 4, pp. 765-772, 1985.
- [30] X. Gao, P.M. Lallican and C. Viard-Gaudin, "A two-stage online handwritten Chinese character segmentation algorithm based on dynamic programming," *Proceedings of the 8th International Conference on Document Analysis and Recognition(ICDAR)*, Seoul, Korea, pp. 735-739, 2005.
- [31] Q. Fu, X.Q. Ding, T. Liu, Y. Jiang and Z. Ren, "A novel segmentation and recognition algorithm for Chinese handwritten address character strings," *Proceedings of the 18th International Conference on Pattern Recognition*, pp. 974-977, 2006.
- [32] Q.F. Wang, F. Yin and C.L. Liu, "Handwritten Chinese text recognition by integrating multiple contexts," *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 34, no. 8, pp. 1469-1481, 2012.
- [33] S. Senda and K. Yamada, "A maximum-likelihood approach to segmentation-based recognition of unconstrained handwriting text," *Proceedings of the 6th International Conference on Document Analysis and Recognition*, Seattle, USA, pp. 184-188, 2001.
- [34] N.X. Li and L.W. Jin, "A Bayesian-based method of unconstrained handwritten offline

Chinese text Line recognition,” *International Journal on Document Analysis and Recognition*, vol. 16, no. 1, pp. 17-31, 2013.

- [35] R.O. Duda, P.E. Hart and D.G. Stork, “Pattern classification,” John Wiley & Sons, 2012.
- [36] X.D. Zhou, C.L. Liu, and M. Nakagawa, “Online handwritten Japanese character string recognition using conditional random fields,” *Proceedings of the 10th International Conference on Document Analysis and Recognition*, Barcelona, Spain, pp. 521-525, 2009.
- [37] X.D. Zhou, J.L. Yu, C.L. Liu, T. Nagasaki and K. Marukawa, “Online handwritten Japanese character string recognition incorporating geometric context,” *Proceedings of the 9th International Conference on Document Analysis and Recognition*, Curitiba, Brazil, pp. 48-52, 2007.
- [38] T.M. Breuel, “A system for the off-line recognition of handwritten text,” *Proceedings of the 12th International Conference on Pattern Recognition*, vol. 2, Jerusalem, Israel, pp. 129-134, 1994.
- [39] M.S. Kim, S. Ryu, K.T. Cho, T.H. Rhee, H.I. Choi and J.H. Kim, “Recognition-based digitalization of Korean historical archives,” *Proceedings of the Asia Information Retrieval Symposium*, Beijing, China, pp. 281-288, 2004.
- [40] Q. Fu, X.Q. Ding, C.S. Liu, Y. Jiang and Z. Ren, “A hidden Markov model based segmentation and recognition algorithm for Chinese handwritten address character strings,” *Proceedings of the 8th International Conference on Document Analysis and Recognition*, Seoul, Korea, pp. 590-594, 2005.
- [41] X.Q. Ding and H.L. Liu, “Segmentation-driven offline handwritten Chinese and Arabic script recognition,” *Proceedings of the Summit on Arabic and Chinese Handwriting*, College Park, USA, pp. 61-73, 2006.
- [42] S. Tulyakov and V. Govindaraju, “Probabilistic model for segmentation based word recognition with lexicon,” *Proceedings of the 6th International Conference on Document Analysis and Recognition*, Seattle, USA, pp. 164-167, 2001.
- [43] B. Zhu, J. Gao and M. Nakagawa, “Objective function design for MCE-based combination of on-line and off-line character recognizers for on-line handwritten Japanese text recognition,” *Proceedings of the 11th International Conference on Document Analysis and Recognition*, Beijing, China, pp. 594-598, 2011.
- [44] J. Gao, B. Zhu and M. Nakagawa, “Development of a robust and compact on-line handwritten Japanese text recognizer for hand-held devices,” *IEICE Trans. Inf. & Sys.*, vol.

E96-D, no. 4, pp. 927-938, 2013.

- [45] J. Gao, B. Zhu and M. Nakagawa, "Building compact recognizer with recognition rate maintained for on-line handwritten Japanese text recognition," *Pattern Recognition Letters*, vol. 35, pp. 169-177, 2014.
- [46] D.H. Wang, C.L. Liu and X.D. Zhou, "An approach for real-time recognition of online Chinese handwritten sentences," *Pattern Recognition*, vol. 45, no. 10, pp. 3661-3675, 2012.
- [47] T.H. Su, T.W. Zhang and D.J. Guan, "Corpus-based HIT-MW database for offline recognition of general-purpose Chinese handwritten text," *International Journal of Document Analysis and Recognition*, vol. 10, no. 1, pp. 27-38, 2007.
- [48] C.L. Liu, F. Yin, D.H. Wang, and Q.F. Wang, "CASIA Online and Offline Chinese Handwriting Databases," *Proceedings of the 11th International Conference on Document Analysis and Recognition*, Beijing, China, pp. 37-41, 2011.
- [49] B.H. Juang, W. Hou and C.H. Lee, "Minimum classification error rate methods for speech recognition," *IEEE Trans. Speech and Audio Processing*, vol. 5, no. 3, pp. 257-265, 1997.
- [50] H. Robbins and S. Monro, "A stochastic approximation method," *The Annals of Mathematical Statistics*, pp. 400-407, 1951.
- [51] H. Ney and S. Ortman, "Progress in dynamic programming search for LVCSR," *Proceedings of the IEEE*, vol. 88, no. 8, pp. 1224-1240, 2000.
- [52] C.L. Liu, M. Koga and H. Fujisawa, "Lexicon-driven segmentation and recognition of handwritten character strings for Japanese address reading," *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 24, no. 11, pp. 1425-1437, 2002.
- [53] B. Zhu and M. Nakagawa, "Trie-lexicon-driven recognition for on-Line handwritten Japanese disease names using a time-synchronous method," *Proceedings of the 11th International Conference on Document Analysis and Recognition*, Beijing, China, pp. 1130-1134, 2011.
- [54] Y. Chiang and C. Knoblock, "Recognition of Multi-oriented, Multi-sized, and Curved Text", *Proc. 11th ICDAR*, pp. 18-21, 2011.
- [55] N. Sun, M. Abe, and Y. Nemoto, "A Handwritten Character Recognition System by Using improved Directional Element Feature and Subspace Method (in Japanese)," *IEICE Trans. Inf. & Sys.*, vol. J78-D-II, no. 6, pp. 922-930, 1995.
- [56] G. Srikantan, D.S. Lee and J.T. Favata, "Comparison of normalization methods for character recognition, *Pro. of the third International Conference on Document Analysis and*

Recognition,” Montreal, Canada, pp. 719–722, 1995.

- [57] J.O. Jr, L.R. Veloso, J.M. de Carvalho, “Interpolation/decimation scheme applied to size normalization of characters images,” Proc. of the 15th International Conference on Pattern Recognition, Barcelona, Spain, vol. 2, pp. 577–580, 2000.
- [58] C.L. Liu, M. Koga, H. Sako and H. Fujisawa, “Aspect ratio adaptive normalization for handwritten character recognition,” Lecture Notes in Computer Science, vol. 1948, pp. 418-425, 2000.
- [59] U. Ramer, “An Iterative Procedure for the Polygonal Approximation of Plan Closed Curves,” Computer Graphics and Image Processing, vol. 1, pp.244-256, 1972.
- [60] S. Z. Li, “Markov Random Field Modeling in Image Analysis,” Springer, ISBN: 978-1-848000-278-4, 2009.
- [61] J. Zeng and Z.Q. Liu, “Markov Random Fields for Handwritten Chinese Character Recognition,” Proc. 8th International Conference on Document Analysis and Recognition, Seoul, pp. 101–105, 2005.
- [62] X.D. Zhou and C.L. Liu, “Text/non-text Ink Stroke Classification in Japanese Handwriting Based on Markov Random Fields,” Proc. of the 9th International Conference on Document Analysis and Recognition, Curitiba, Brazil, pp. 377-381, 2007.
- [63] S.J. Cho, J.H. Kim, “Bayesian Network Modeling of Strokes and their Relationships for On-line Handwriting Recognition,” Pattern Recognition, vol. 37, pp.253-264, 2004.
- [64] J. Lafferty, A. McCallum, and F. Pereira, “Conditional Random Fields: Probabilistic Models for Segmenting and Labeling Sequence Data,” Proc. of the 18th ICML, pp. 282-289, 2001.
- [65] B.H. Juang and S. Katagiri, “Discriminative Learning for Minimum Error Classification,” IEEE Trans. Signal Processing, vol.40, no.12, pp.3043-3054, 1992.
- [66] S. Shetty, H. Srinivasan and S. Srihari, “Handwritten Word Recognition Using Conditional Random Fields,” Proc. of the 9th ICDAR, pp.1098-1102, 2007.
- [67] X.D. Zhou, C.L. Liu, and M. Nakagawa, “Online Handwritten Japanese Character String Recognition Using Conditional Random Fields,” Proc. of the 10th International Conference on Document Analysis and Recognition, Barcelona, Spain, 2009.
- [68] M. Liwicki and H. Bunke, “HMM-based On-line Recognition of Handwritten Whiteboard Notes,” Proc. 10th International Workshop on Frontiers in Handwriting Recognition (IWFHR), pp. 595-599, 2006.

- [69] Y. Katayama, S. Uchida and H. Sakoe, "HMM for On-Line Handwriting Recognition by Selective Use of Pen-Coordinate Feature and Pen-Direction Feature (in Japanese)," *IEICE Trans. Inf. & Sys.*, vol. J91-D, no.8, pp. 2112-2120, 2008.
- [70] G. Nagy, N. Tuong, "Normalization techniques for hand printed numerals," *Scientific Application, Communication of the ACM* 13th, no.8, pp. 475-481, 1970.
- [71] R.G. Casey, "Moment normalization of Hand-printed character," *IBM J. Res. Dev.* 14, pp.548-557, 1970.
- [72] J. Tsukumo, H. Tanaka, "Classification of Hand Printed Chinese Characters Using Nonlinear Normalization and Correlation Methods," *Proc. of the 9th International Conference on Pattern Recognition*, Rome, Italy, pp. 168-171, 1988.
- [73] H. Yamada, K. Yamamoto and T. Saito, "A Nonlinear Normalization Method for Hand Printed Kanji Character Recognition Based on Line Density Equalization," *Pattern Recognition*, vol. 23, no.9, pp. 1023-1029, 1990.
- [74] A.D.S. Britto JR. and R. Sabourin, "Improvement in Handwritten Numeral String Recognition by Slant Normalization and Contextual Information," *Proc. of the 7th International Workshop on Frontiers of Handwriting Recognition*, Amsterdam, Netherlands, pp. 323-332, 2000.
- [75] C.L. Liu and K. Nakashima, "Handwritten Digit Recognition: Benchmarking of State-of-the-Art Techniques," *Pattern Recognition*, vol. 36, no.10, pp. 2271-2285, 2003.
- [76] Y. Yamashita and K. Higuchi., "Classification of Hand-printed Kanji Characters by the Structured Segment Matching Method, *Pattern Recognition Letters*, vol. 1, pp. 475-479, 1983.
- [77] C.L. Liu, "Preprocessing and statistical/structural feature extraction for handwritten numeral recognition," *Academic Press, London*, pp. 161-168, 1997.
- [78] G. Jun, N. Sun, Y. Nemoto, M. Kimura, H Echigo, "Recognition of Handwritten Characters Using Pattern Transformation Method with Cosine Function," (in Japanese) *IEICE Trans. Inf. & Sys.*, vol.J76-D-II, no.4, pp. 835-842, 1993.
- [79] R. J. Carroll and D. Ruppert, "On prediction and the power transformation family," *Academic Press, London* , 1997.
- [80] T. Wakabayashi, "On the Size and Variable Transformation of Feature Vector for Handwritten Character Recognition," *IEICE Trans. Inf. & Sys.*, Japan vol. J76-D-II no.12, pp. 2495-2503, 1993.
- [81] Heiden R.Van Der and F.C.A. Groen, "The Box-Cox Metric for Nearest Neighbor

Classification Improvement,” *Pattern Recognition*, vol. 30, no.2, pp. 273–279, 1997.

- [82] F. Kimura, K. Takashina, S. Tsuruoka and Y. Miyake, “Modified quadratic discriminant functions and the application to Chinese character recognition,” *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 9, no. 1, pp. 149-153, 1987.
- [83] M. Nakagawa and M. Onuma, “Online handwritten Japanese text recognition free from constrains on line direction and character orientation,” In *Proceedings of International Conference on Document Analysis and Recognition(ICDAR)*, pp.519–523, Edinburgh, Scotland, 2003.
- [84] F. Yin and C.L. Liu. “Handwritten Chinese text line segmentation by clustering with distance metric learning. *Pattern Recognition*, vol.42, no.12, pp.3146–3157, 2009.
- [85] M. Liwicki, E. Indermuhle, and H. Bunke. “On-line handwritten text line detection using dynamic programming,” In *Proceedings of International Conference on Document Analysis and Recognition (ICDAR)*, vol. 1, pp. 447–451, Curitiba, Brazil, 2007.
- [86] M. Shilman, Z. Wei, S. Raghupathy, P. Simard, and D. Jones. “Discerning structure from freeform handwritten notes,” In *Proceedings of International Conference on Document Analysis and Recognition (ICDAR)*, pp. 60–65, Edinburgh, Scotland, 2003.
- [87] X.D. Zhou, D. H. Wang and C.L. Liu, “A robust approach to text line grouping in online handwritten Japanese documents,” *Pattern Recognition*, vol. 42, no. 9, pp. 2077-2088, 2009.
- [88] Y. Nievergelt. “Total least squares: State-of-the-art regression in numerical analysis. *SIAM review*,”vol. 36, no.2, pp. 258–264, 1994.
- [89] V.N. Vapnik, “*The Nature of Statistical Learning Theory*,” John-Wiley Press, ISBN 978-1-4419-3160-3,1998.
- [90] Y. Sun, T.S. Butler, A. Shafarenko, R. Adams, M. Loomes and N. Davey, “Segmenting handwritten text using supervised classification techniques,” *Proceedings of IEEE International Joint Conference on Neural Networks*, vol. 1, pp. 657-662, 2004.
- [91] Z. Harbi, Y. Hicks, R. Setchi and A. Bayer, “Segmentation of Clock Drawings Based on Spatial and Temporal Features,” *Procedia Computer Science* 60, pp. 1640-1648, 2015.
- [92] T. Joachims, “*Making large-scale SVM learning practical*”, Universität Dortmund, 1999.
- [93] J. Gao, B. Zhu and M. Nakagawa, “Development of a Robust and Compact On-Line Handwritten Japanese Text Recognizer for Hand-Held Devices,” *IEICE Trans. Inf. & Syst.*, vol. E96-D, no.4, pp. 927-938, 2013.

- [94] C.L. Liu and K. Marukawa, "Handwritten numeral string recognition: character-level vs. string-level classifier training," Proceedings of the 17th International Conference on Pattern Recognition, Cambridge, UK, vol. 1, pp. 405-408, 2004.
- [95] C.Y. Suen, "N-gram statistics for natural language understanding and text processing," IEEE Trans. Pattern Analysis and Machine Intelligence, vol. 1, no. 2, pp. 164-172, 1979.
- [96] K. Takeuchi and Y. Matsumoto, "Japanese OCR correction using stochastic morphological analyzer and probabilistic N-gram word model," International Journal of Computer Processing of Languages, vol. 13, no. 1, pp. 69-82, 2000.
- [97] P.K. Wong, and C. Chan, "Post processing statistical language models for handwritten Chinese character recognizer," IEEE Trans. Systems, Man, and Cybernetics, Part B: Cybernetics, vol. 29, no. 2, pp. 286-291, 1999.
- [98] N. Iwayama and K. Ishigaki, "Adaptive context processing in on-line handwritten character recognition," Proceedings of the 7th International Workshop Frontiers in Handwriting Recognition, pp. 469-474, 2000.
- [99] A. Vinciarelli, S. Bengio and H. Bunke, "Offline recognition of unconstrained handwritten texts using HMMs and statistical language models," IEEE Trans. Pattern Analysis and Machine Intelligence, vol. 26, no. 6, pp.709-720, 2004.
- [100] Q.F. Wang, F. Yin and C.L. Liu, "Unsupervised language model adaptation for handwritten Chinese text recognition," Pattern Recognition, vol. 47, no. 3, pp. 1202-1216, 2014.
- [101] N. Li, J. Chen, H. Cao, B. Zhang and P. Natarajan, "Applications of Recurrent Neural Network Language Model in Offline Handwriting Recognition and Word Spotting," Proceedings of the 14th International Conference on Frontiers in Handwriting, Crete, Greece, pp. 134-139, 2014.
- [102] S.F. Chen and J. Goodman, "An empirical study of smoothing techniques for language modeling," Computer Speech and Language, vol. 13, pp. 359-394, 1999.
- [103] F. Yin, Q.F. Wang, and C.L. Liu, "Transcript mapping for handwritten Chinese documents by integrating character recognition model and geometric context," Pattern Recognition, vol. 46, no. 10, pp. 2807-2818, 2013.
- [104] Y.C. Wu, F. Yin and C.L. Liu, "Evaluation of Geometric Context Models for Handwritten Numeral String Recognition," Proceedings of the 14th International Conference on Frontiers in Handwriting Recognition, Crete, Greece, pp. 193-198, 2014.

- [105] M. Nakagawa and K. Matumoto, "Collection of on-line handwritten Japanese character pattern database and their analysis," *Int. J. Document Anal Recognition*, vol. 7, no. 1, pp. 69-81, 2004.

Appendix- I : List of Figure

Figure2- 1	Segmentation lattice. (SP is segmentation point and UP is undecided point.)	16
Figure3- 1	Process of feature points extraction	27
Figure4- 1	Character orientation and line direction.....	41
Figure4- 2	Text line element, Character orientation and line direction.	41
Figure4- 3	Recognition steps in the system.....	42
Figure4- 4	Flow chart of text line segmentation process.....	43
Figure4- 5	Line segmentation - block grouping.	44
Figure4- 6	Line segmentation - pre-segmentation.....	45
Figure4- 7	Line segmentation - temporal segmentation.....	45
Figure4- 8	Segmentation candidate lattice of a text line string with six blocks(six straight edges). Each internal node corresponds to an off-stroke between blocks. The curved edges denote text lines comprising multiple blocks.	46
Figure4- 9	An example of text line segmentation.	47
Figure4- 10	Grouping process. The grouping results are bounded with red rectangular boxes: (a) stroke classification results,(Blue: text stroke; gray: non-text strokes.) (b) pre-segmentation results, (c) temporal segmentation results, and (d)temporal merge results. 47	47
Figure4- 11	Flow processing of character orientation.....	48
Figure4- 12	Pen movement	49
Figure4- 13	Two main peaks in pen movement direction.	49
Figure4- 14	Estimation method of character orientation.....	50

Figure4- 15	Normalization of character orientation.	50
Figure4- 16	Quantization of line direction.	51
Figure4- 17	A quasi-program to obtain the feature fd	52
Figure4- 18	Distance feature fd for line direction R	52
Figure4- 19	A quasi-program to obtain the feature fi	53
Figure4- 20	Features to obtain fi for line direction R	53
Figure4- 21	Setting thresholds for hypothetical segmentation[8].	54
Figure4- 22	Example of Support vectors.....	58
Figure4- 23	Segmentation-recognition candidate lattice.....	61
Figure4- 24	Geometric features.....	63
Figure4- 25	Inner gap feature within a character pattern.	63
Figure5- 1	Examples of character-based (a) and word-based (b) unigram, bigram and trigram model.	68
Figure5- 2	Geometric features.....	74
Figure5- 3	Inner gap feature within a character pattern	74
Figure6- 1	Examples in <i>HANDS-Kondate_t_bf-2001-11</i> database from Page23 to Page29.	78
Figure6- 2	Page of 23-28 and the result of reognition	84
Figure6- 3	Examples of rotated handwritten text lines.....	85
Figure7- 1	Examples of recognition errors.....	89

Appendix- II : List of Tables

Table3- 1	Mapping of linear normalization.	26
Table4- 1	Terms to obtain 21 features.	59
Table4- 2	21 features for over-segmentation extracted from each off-stroke.	60
Table6- 1	Pages in <i>HANDS-Kondate_t_bf-2001-11</i> (100 people).	76
Table6- 2	Number of samples in training and testing sets for each direction.	78
Table6- 3	Number of samples in training and testing sets for each direction (dataset1) ..	79
Table6- 4	Number of samples in training and testing sets for each direction (dataset2) ..	79
Table6- 5	Number of samples in training and testing sets for each direction (dataset3) ..	79
Table6- 6	Number of samples in training and testing sets for each direction (dataset4) ..	80
Table6- 7	Performance on reduced dimensionalities.	81
Table6- 8	Performance optimized by GA and MCE-SGD v.s segmentation-updated system.	82
Table6- 9	Performance on mixture of vertical, horizontal and slanting lines.	83
Table6- 10	Performance on rotated horizontal/vertical handwritings.	86
Table6- 11	Time complexity.	86
Table6- 12	Memory requirement for dictionaries.	87

Appendix-III: Author's Publications

Journal Paper

- [1] Yuechan Hao, Bilan Zhu and Masaki Nakagawa, "A Line-direction-free and Character-orientation-free On-line Handwritten Japanese Text Recognition System," IEICE Trans. Inf. & Syst. Vol. E99-D, no.1, pp. 197-207, 2016.

International Conference Paper

- [2] Yuechan Hao, Bilan Zhu and Masaki Nakagawa, "Large Improvement in Line-direction-free and Character-orientation-free On-line Handwritten Japanese Text Recognition," Proc. 14th ICFHR, pp. 329-334, 2014.