Ink Matching of Cursive Chinese Handwritten Annotations

Daniel P. Lopresti *Matthew Y. Ma**
Panasonic Information and Networking Technology Lab.
Panasonic Technologies, Inc.
2 Research Way, Princeton, NJ 08540

Patrick S.P. Wang, IAPR Fellow
College of Computer Science
Northeastern University, Boston, MA 02115

Jill D. Crisman
Department of Electrical and Computer Engineering
Northeastern University, Boston, MA 02115

Abstract

In this paper, we discuss the notion of treating electronic ink as first class data without attempting to recognize it by presenting two different variations of approximate ink matching (AIM) for searching ink data. We also illustrate a pen-based electronic document annotating and browsing system and methods for searching handdrawn personal notes employing the described matching schemes. Adapting from the *Learning by knowledge* paradigm, we propose a semantic matching network that applies semantics of Chinese language early in the process of ink matching. Finally we evaluate several key components in our entire ink matching network via experiments. Preliminary experimental results show the approximate ink matching algorithms perform well, despite the informal and highly variable nature of Chinese handwriting. Our experiments also show some promising results on semantic matching and the feasibility of our semantic matching architecture.

Keywords: Approximate ink matching, Semantic matching, Electronic ink matching, Chinese handwritten annotations, Radical extraction

1 Introduction

In Chinese office automation systems, the input of Chinese characters into the computer is a bottleneck. It has been reported that efficient input methods for Chinese characters were

*To whom correspondence should be made. E-mail: mma@research.panasonic.com

proposed and utilized [19], and more than 100 Chinese characters per minute can be input using a keyboard. However, these input methods are mostly for trained typists; they are too complicated for an ordinary person to learn.

With the advancement of computer technology and input devices such as the stylus, the next generation of pen computers and PDA's have a lot of potential. Handwriting recognition, however, has proved to be a more difficult problem than most people first expected. The use of stylus in a pen computing system brings the advantages in many ways, it need not be limited to input for handwriting recognition as this is a very difficult task especially there is a wide variation in the way people write. Nakagawa *et al.* [14] presented the concept of lazy recognition which is the notion that constantly presenting recognition results to the user (often with errors that the user is forced to correct) interferes with the creative process of writing. Even if HWX accuracy were 100%, it still might bother the user to replace his/her handwriting with text as it is recognized.

Along these lines, it is possible to do handwriting recognition completely in the background (without showing the results to the user) and then use this for retrieval. However, one potential problem with this approach is that the user receives no immediate feedback, and the handwriting recognition might be doing a very bad job and the user will not realize this until he/she tries to do retrieval.

Instead of handwriting recognition, some research have been done around electronic ink matching, including Lopresti and Tomkins's earlier work [8]. Similar research can also be found in the work of Poon *et al.* [16] and Reynolds *et al.* [17], but no applications were identified. Manmatha *et al* has done some work in handwriting matching [10], but mainly on scanned (off-line) images instead of electronic (on-line) ink. Pavlidis *et al* [15] has used shape metamorphosis to recognize on-line handwritten patterns, but this is limited to singly connected shapes.

Our work around electronic ink matching is based on the notion of "treating handdrawn annotations as first class" data. The idea is to take electronic ink as a temporal sequence of pen strokes – without attempting to recognize it. Efficient search algorithms based on approximate string matching are then used for matching query ink sequence with the previously stored ink sequences in a database. This idea was later extended to searching lengthy handwritten notes for keywords ("word spotting").

In this paper, we explore the application of these ideas to searching Chinese handwritten annotations in a pen-based document browser. Then, we describe the approximate ink matching (AIM) algorithm that is being used in the application. Using such an approach, some of the difficult tasks encountered in handwriting recognition such as low accuracy rates due to the variation in writing styles can be avoided.

While our AIM algorithm matches handwriting based on strokes without trying to recognize it, a human, on the other hand, might take a different approach. As illustrated in Figure 1, when human tries to match two pieces of handwriting, he/she first tries to identify semantics i.e. radicals in Chinese language, then match these semantics. Radicals are small structural parts that form a character. They are basic elements of semantics in Chinese, and they usually have their own meanings.

Based on this concept, we extended our ink matching algorithm to be able to identify radicals in the Chinese language that is contained in handwriting. These extracted radicals

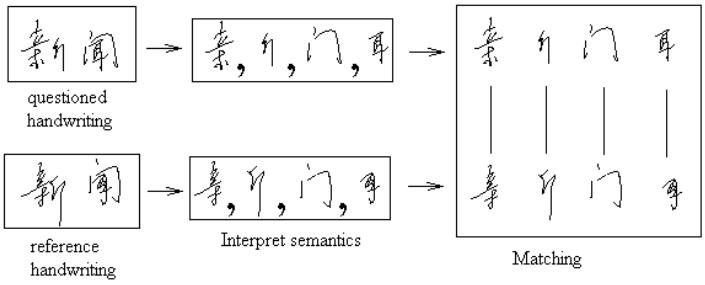

Figure 1: Use of semantics when matching two pieces of handwriting in Chinese.

will then be used to "classify" the entries in the annotation database. When a query is entered by the user, only items with relevant radicals in the database will be searched, thus speeding up the existing ink matching by reducing the size of the problem. By identifying the radicals in handdrawn Chinese annotations, we can also convert ink into a sequence of computer codes that represent basic elements of Chinese language - radicals. This process does not involve a large vocabulary and hence is far less complicated than traditional HWX, but can be suitable for our application of matching ink annotations. This matching scheme is called *semantic matching*.

Semantic matching employs the same concept of extracting radicals used by most handwriting recognition techniques, but it is not doing handwriting recognition. Semantic matching is an encouraging idea in improving existing ink matching for Chinese handwriting especially cursive handdrawn annotations. It is also a good example of *Learning by knowledge* paradigm [23] in processing Chinese handwriting, in which accumulated knowledge is being iteratively used.

The remainder of this paper is organized as follows. In Section 2, we provide an overview of the Chinese ink annotation application. Section 3 describes in detail our approximate ink matching (AIM) algorithm. Section 4 describes our ongoing research in semantic matching and illustrates some interesting results. Section 5 discusses a systematic procedure for evaluating the performance of ink matching. Finally, in Section 6, we give our conclusions.

2 Overview of the Ink Annotation System

Annotation is a useful and important operation for handling documents. A quick survey of the desk of an average office worker will turn up many pieces of paper that contain handwritten notes, items that have been circled to indicate their importance, text that has been "edited" by crossing it out, etc. Clearly, any system that attempts to duplicate the familiarity of paper-based document handling must include provisions for user-entered annotations. Mostly important, these annotations should be searchable.

Unfortunately, the search options provided in current document management systems are mostly text-based. Usually text-based descriptions of the documents are either extracted from the scanned documents using document image analysis and OCR techniques,

or entered manually by the user using a keyboard. Such search techniques cannot be used for handdrawn annotations created by non-text based input devices, such as a pen.

The text-based search technique itself also has some limitations in real applications. For example, developing similar systems for Chinese-based documents proves to be a challenge. Unlike most western languages, Chinese does not have a fixed alphabet set. Instead, it has a much larger character set than ASCII. Consequently, entering Chinese characters into a computer is a time-consuming task since no simple keyboard is available. Also, the large character set makes Chinese character recognition, either printed or handwritten, more difficult than, say, English OCR.

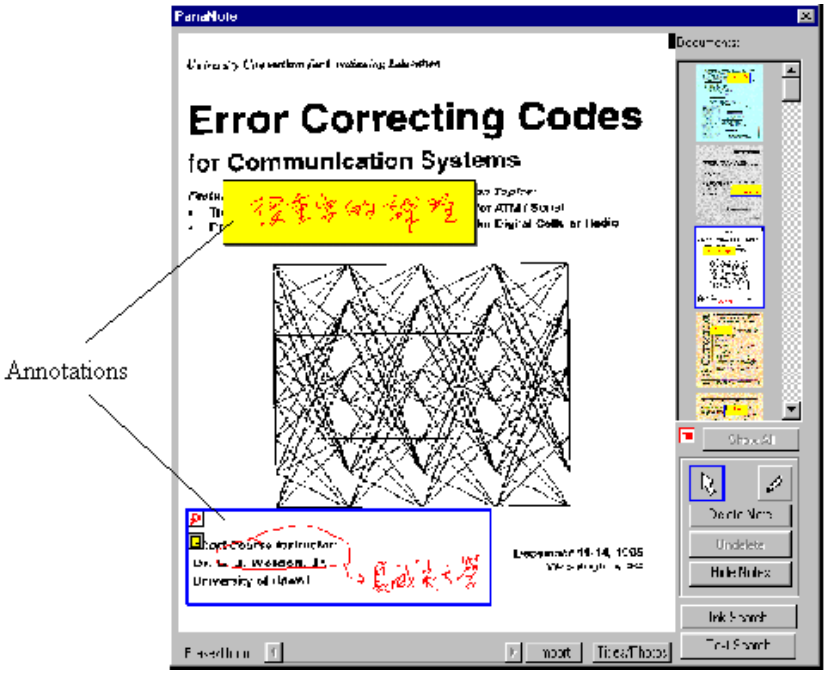

Figure 2: Annotations are written directly on a document.

In this section, we describe an ink annotation tool that is advantageous over a keyboard in a document system, this is particularly important for a Chinese document system which a Chinese keyboard is hard to implement. Our goal is to give simplicity and ease of use when users write thus even someone with little computer background should feel comfortable using our software as writing annotations on paper documents.

As illustrated in Figure 2, scanned documents can be browsed and annotations can be written directly on a document. Once the annotation is written, it can be highlighted, moved, hidden, deleted or changed to a "stick-on" note with yellow background. Multiple annotations can exist in the same document page, each having their own mode: hidden or displayed to the user. Ink editing capability is also integrated in the annotation control panel.

The annotation software also provides a search tool which allows users to write a query and search for a previously written annotation. The search algorithm (AIM) we use is based

on the assumption that people write differently, but they normally write in a consistent way. As illustrated in Figure 3, four users were asked to write the same annotation twice, at different times. By looking at one column, we can see that four people write differently; looking at each row, we can see that each user writes consistently.

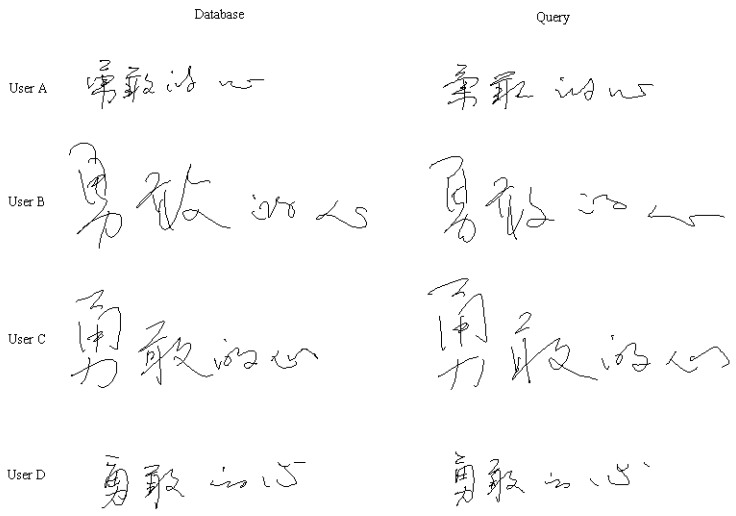

Figure 3: Handwriting is highly personal. It does, however, exhibit significant self-consistency.

The AIM uses a fuzzy search technique. Unlike exact match, AIM never rejects a search query. It always gives results by sorting them so the best match is shown first and the worst is shown last. The user interface (as shown in Figure 4) is provided to let users go through each annotation in the order of rank. Therefore, if the best match is not the expected match, users can still see the rest of top hits at once so they can quickly decide which one is exactly what they want. Clicking on the selected annotation will display the associated document (i.e. the document that this annotation is written on) to the user.

As seen in Figure 4, an ink query for "important" in Chinese yields matches on "important news", "very important", "a very important course", "important meeting" etc. This is called word spotting. The word spotting capability in AIM allows subsequence matching - a shorter query to be matched to a region contained in a longer database entry. Again, this capability is not just limited to languages, it can also be used to match graphical annotations. Figure 5 illustrates the word spotting concept. The word spotting in AIM has the benefit of retrieving not only the desired annotation itself, but also related annotations.

3 Approximate Ink Matching

The approximate ink matching algorithm (AIM) that was used in the above described ink annotation system was developed to perform matching at the pen-stroke level. This

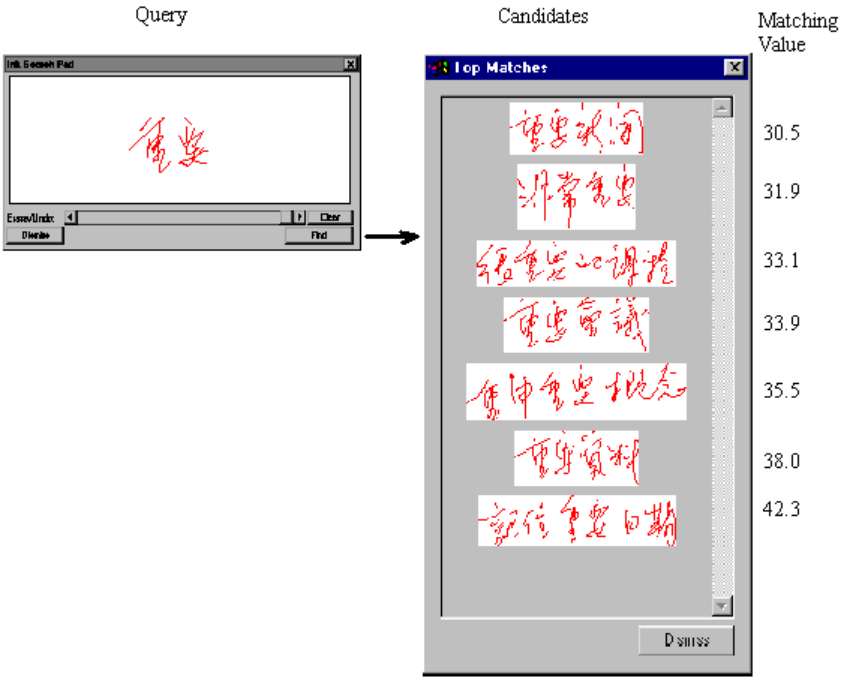

Figure 4: A ranked list of top matches is returned to the user.

approach has the advantage of allowing user to do well against a broad range of handwriting, including some so poor that it is effectively illegible. The same algorithm can be used in both English and Chinese, irregardless of source language. The algorithm applies dynamic programming with a recurrence similar to that used for string edit distance, but with a different set of operations and costs. The top-level organization of the ink search algorithm is shown in Figure 6. The details of the algorithm will be described in the following.

Stroke segmentation

Ink consists of a sequence of points representing the movement of stylus when the user is writing. First, the incoming points are grouped into strokes. Currently we break each stylus

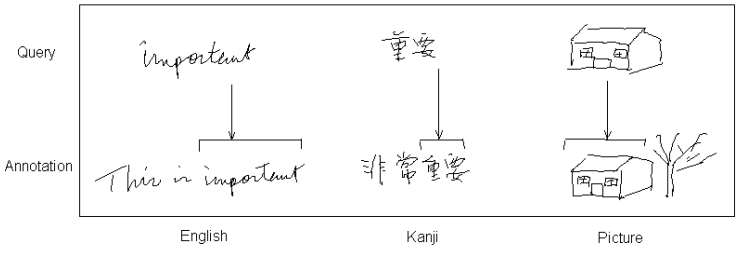

Figure 5: "Word" spotting (subsequence matching).

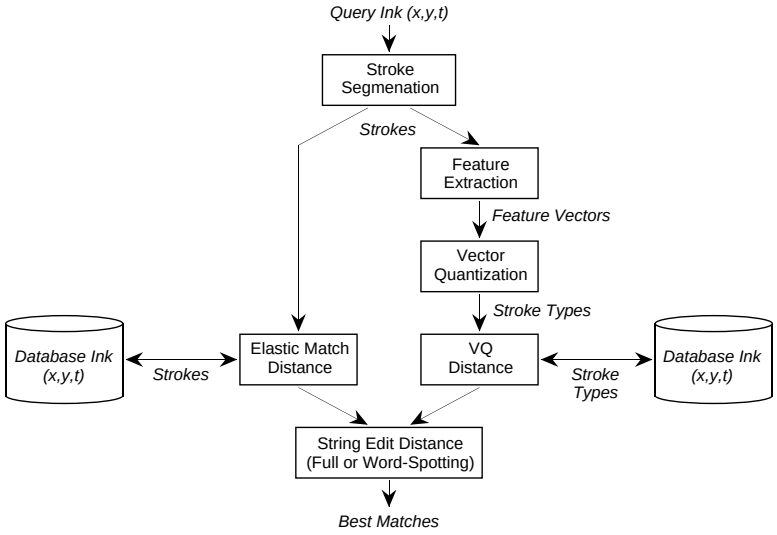

Figure 6: Overview of the ink search algorithm.

movement from "pen-down" to "pen-up" at local minima of the y values and segment into strokes.

Elastic distance/VQ distance

There are two variations of the algorithm as shown in Figure 6. One is elastic matching which is based directly on segmented strokes. The elastic matching works directly with raw ink stroke thus does not need training. The other is VQ matching which extracts 13 features used by Rubine in the context of gesture recognition [18] from each stroke and forms a 13-dimensional feature vector, then classifies the feature vectors into one of 64 different stroke types. From then on, the ink sequences are represented as strings of stroke types. This particular feature set seems to do well at discriminating single strokes and can be updated efficiently as new points arrive. The VQ codebook was computed from a small sample of handwriting and the same VQ codebook can be used for all users. The details of vector quantization on strokes were described in our earlier work [8][7]. The cost functions for the edit distance, which will be described later, is then calculated based on the segmented strokes (elastic) or classified stroke types (VQ).

Edit distance

In the last phase, we compute the edit distance between the stroke sequence associated with the query ink and the pre-computed sequence for the reference ink. Let $S = s_1 s_2 ... s_m$ and $T = t_1 t_2 ... t_n$ be two symbols (in our case ink sequences) where s_i and t_i represent strokes, and let $d(S, T)$ be the minimum cost of any sequence of basic operations that transforms S

into T. Solve $d(S,T)$ can be implemented by a dynamic recurrence:

$$d_{i,j} = \min \begin{cases} d_{i-1,j} & + & c_{del}(s_i) \\ d_{i,j-1} & + & c_{ins}(t_j) \\ d_{i-1,j-1} & + & c_{sub}(s_i, t_j) \end{cases} \qquad 1 \le i \le m,\ 1 \le j \le n \qquad (1)$$

where $d_{i,j}$ is distance between the first i symbols of stroke sequence S and the first j symbols of stroke sequence T. Note that $d(S,T) = d_{m,n}$. c_{del}, c_{ins}, and c_{sub} are costs associated with three basic operations as in the traditional case [22]: 1) delete a stroke; 2) insert a stroke; 3) substitute one stroke for another.

In implementation, two operations, split and merge were added in VQ algorithm to account for imperfections in the stroke segmentation algorithm. We build a merge table that contains information of the form "an average stroke of type α merged with an average stroke of type β results in a stroke of type γ". Only two adjacent strokes are considered for merging. The costs for split/merge as well as other system parameters are illustrated in Table 1. The scale parameters w's were from the training process based on small sample set of handwriting in order to obtain normalized operation costs and optimal performance. However, as will be mentioned later in this paper, how to obtain these parameters that work best for Chinese handwriting is one of our future tasks.

Parameters	*Elastic*	*VQ*
c_{del}	w_{del} * length of stroke, $w_{del} = 1$	w_{del} * length of stroke type, $w_{del} = 1.1$
c_{ins}	w_{ins} * length of stroke, $w_{ins} = 1$	w_{ins} * length of stroke type, $w_{ins} = 0.45$
c_{sub}	w_{sub} * Euclidean distance between two strokes, $w_{sub} = 1$	w_{sub} * Mahalanobis distance between two stroke types, $w_{sub} = 1$
c_{merge}	N/A	w_{merge} * length of merged stroke type, $w_{merge} = 2.2$
c_{split}	N/A	The cost of a split from stroke δ to strokes α and $\beta = w_{split}$ * γ, where γ = distance from δ to $merge(\alpha, \beta)$, $w_{del} = 1.1$

Table 1: Cost parameters in approximate ink matching.

Figure 7 shows a detailed trace of our matching on stroke level using elastic ink matching. The upper writing is the query while the lower is the reference database. Each stroke is represented by a bounding box. Strokes that are matched between the query and database are connected by a line. From this figure we can see that every stroke in the query is matched to the correct one in the database.

Approximate ink matching works directly on the raw ink strokes, it can be readily applied to the annotation system we've described in Section 2. As will be described in the next section, approximate ink matching can also be combined with semantics of Chinese

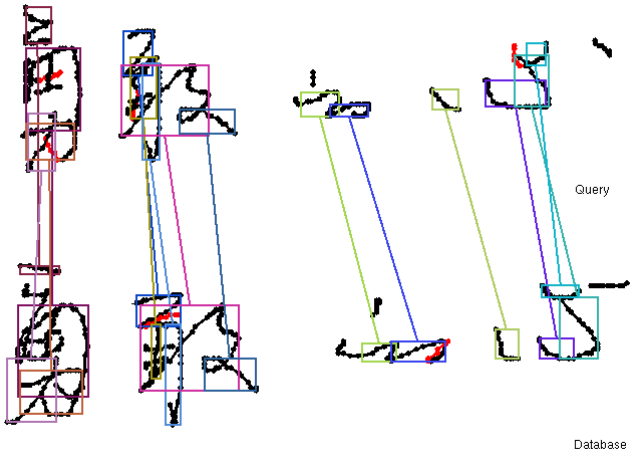

Figure 7: Trace for the edit distance comparison between the query and the database item in question.

language thus construct a semantic matching scheme in order to improve the raw ink-based matching.

4 Semantic Matching

Our work around electronic ink matching is focusing on the user-entered annotations in a document system. This task specially requires dealing with informal cursive handwriting. The structure and semantics of Chinese language make it unique in dealing with problems of matching. The semantics embedded in Chinese plays an important role in human's learning process. On the other hand, semantics should also be employed in the recognition of Chinese characters.

In this section, we discuss the concept of Chinese handwriting matching that uses semantics in a *Learning by knowledge* paradigm. We illustrate that semantics can be used not only in the character recognition or linguistic processing, but also in the early process such as segmentation. Our ongoing research on semantic matching consists of two areas: one is identifying radicals from handwriting and using these radical information to speed up the matching process; another is exploring the matching between handwriting and typed texts via radicals.

4.1 Semantic Matching Scheme

Learning by knowledge is a new methodology presented by Wang [23], in which old, existing knowledge can be accumulated and iteratively used to enforce and help learn new knowledge via a feedback system, as illustrated in Figure 8. Wang has shown that this methodology can be used in understanding and recognition of Chinese characters via a semantic network.

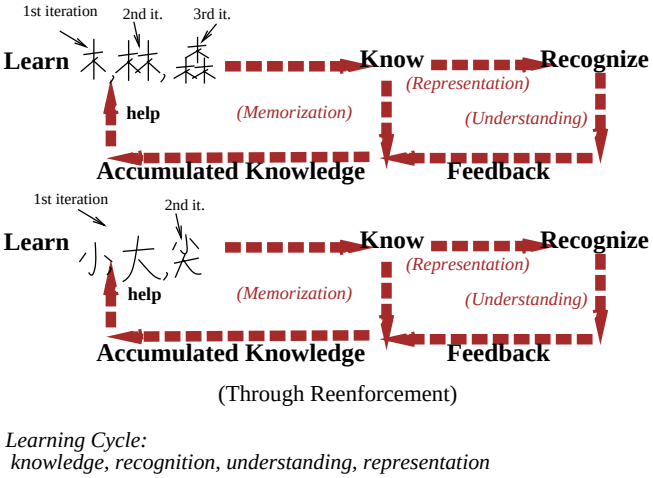

Figure 8: Learning by knowledge in pattern recognition with examples.

We have applied semantics in the *Learning by knowledge* diagram and constructed a semantic matching network as illustrated in in Figure 9. The main steps include pre-processing, character segmentation, radical extraction, classification and matching. As can be seen, our semantic knowledge base was obtained from a learning process (training) and can then be applied in the early processing stage. This would help eliminated unwanted candidates via classification, which in turn speed up the recognition (matching) process. The knowledge learned from semantics may also convert handwriting to computer coded representation of radicals thus make the handwriting searchable by typed text query.

Our method of "learning by knowledge" and "semantic matching" adapted from Wang [23] has advantages over others in the literature, for examples: "learning by rote" according to Ausubel et al [1], of which in a learning process, new information is being added to the process's learning structure without establishing any relationship with the concepts already existing in the knowledge of the learner. A slightly better way of learning was proposed later by Millward [11], known as "meaningful learning" by which new information is roughly related to the relevant concepts already existing in the knowledge structure of the learner.

In the remainder of this section, several details of semantic matching will be described. Pre-processing and character segmentation were described in our previous work [9]. Similar work can also be found elsewhere [20][13].

4.2 Radical extraction

Radicals in Chinese language are small structural parts that form a character. A radical can be a character itself, or can only go with another radical in a character. The extraction of radicals from segmented characters is a non-trivial task. Almost any Chinese handwriting recognition systems have to deal with extraction of radicals or similar information. Some systems [25][26][27][3] [12][4][6] extract feature strokes segments and then form them into radicals. This method would most likely fail on informal cursive Chinese handwriting since

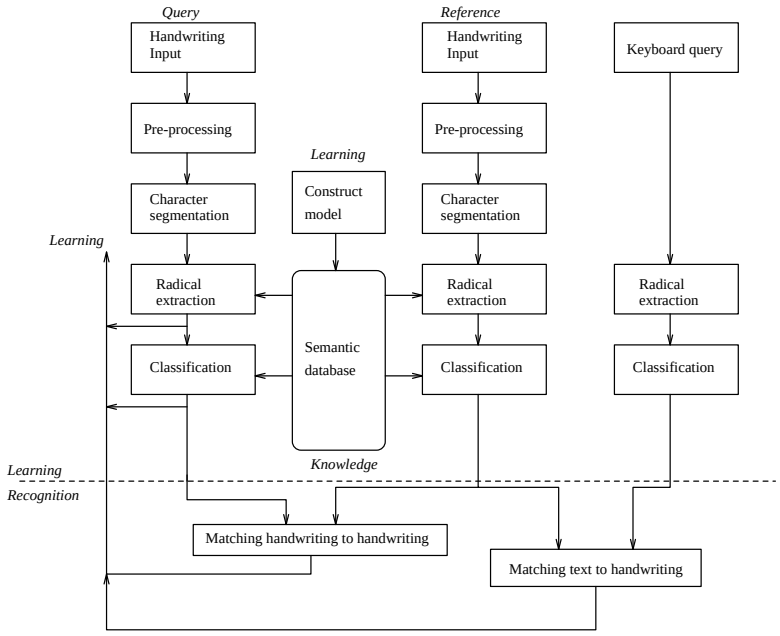

Figure 9: Diagram of semantic matching.

there are so many variations of writing feature stroke in cursive Chinese. Some OCR systems extract radicals based on one assumption that there exist gaps between radicals. This again can not apply to free-format Chinese handwriting. Radicals are often connected together within a character for cursive handwriting thus makes the extraction difficult.

On the other hand, temporal order of strokes are often preserved in on-line handwriting. Almost every Chinese character embeds a particular order of strokes for writing. This order is taught when a child learns how to write. As personal writing styles changes, this temporal order may change. But one person normally write in a consistent way, he/she writes in the same order every time he/she writes the same character. This applies to radicals as well.

The semantic database (also called radical database) consists of a set of basic radicals, which are generated through a training process. The number of samples that need to be collected for training is small since the number of basic radicals in Chinese is very small (200 or so) in compare to the large number of Chinese characters (3000 or so) that are commonly used.

The same elastic matching algorithm (implemented with word spotting) combined with layout detection is used to extract radicals. The word spotting is used to match a radical to a character that contains this radical. However, because radicals contain much fewer strokes than a character or an annotation, this makes the elastic matching less reliable than working on entire annotation. The layout detection of radicals is based on the fact that for a particular radical, it can only reside at a known location within a character. For instance, Cheng and Hsu[2] classify radicals into seven layouts. We divide the bounding box of each character into $3 \times 3 = 9$ equal size regions and calculate the number of stroke points each region contains. As shown in Figure 10, the number of stroke points in each region form a

sequence of 9 numbers. This is called *radical layout profile*.

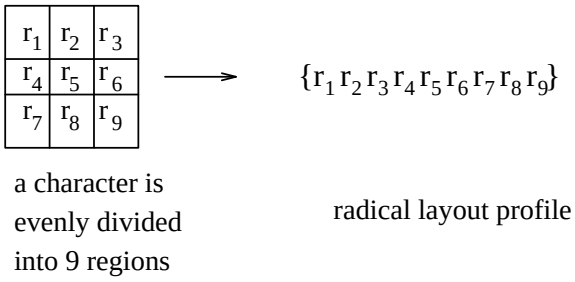

Figure 10: The definition of radical layout profile.

When the semantic database is generated, the layout profile for each radical can be pre-computed. In implementation, every input character is matched partially with limited number of radicals in the semantic database. The returned list of top ranked radicals are then traced back to the original input character in order to compute the layout profile for the radical candidate. The similarity between this newly computed layout profile and the pre-computed profile for the extracting radical is then calculated. Let $P_R = r_1 r_2 r_3 r_4 r_5 r_6 r_7 r_8 r_9$ be the normalized layout profile (by its maximum value) for a radical in the semantic database, and $P_T = t_1 t_2 t_3 t_4 t_5 t_6 t_7 t_8 t_9$ be the normalized profile for traced radical stroke. The similarity measure is defined as:

$$S = c(P_R, P_T) - d(P_R, P_T),$$

whereas $c(P_R, P_T)$ is the correlation expressed as:

$$c(P_R, P_T) = \sum_{i=1}^{9} r_i t_i.$$

and $d(P_R, P_T)$ is the distance thus pays penalty to the overall similarity measure. It is defined as:

$$d(P_R, P_T) = \sum_{i=1}^{9} (r_i - t_i)^2.$$

The similarity values are used to fine tune the order of top ranked extracted radicals. That the distance is greater than correlation yields a negative similarity, and this value will be eliminated in sorting the ranks. Since the number of radicals in the semantic database is limited the matching time on a fast CPU computer will be small. Figure 11 illustrates an example of radical layout profile and its similarity to the profile for radical model.

As shown in Figure 11, when radicals in the semantic database are matched against a questioned character by computing the minimum distance, the elastic matching with word spotting tries to find the best location that the radical fits within the character. This location can be traced back in the ink matching algorithm in order to computer the radical layout profile. With the similarity measure, the correctly located radicals yields a higher value than those not.

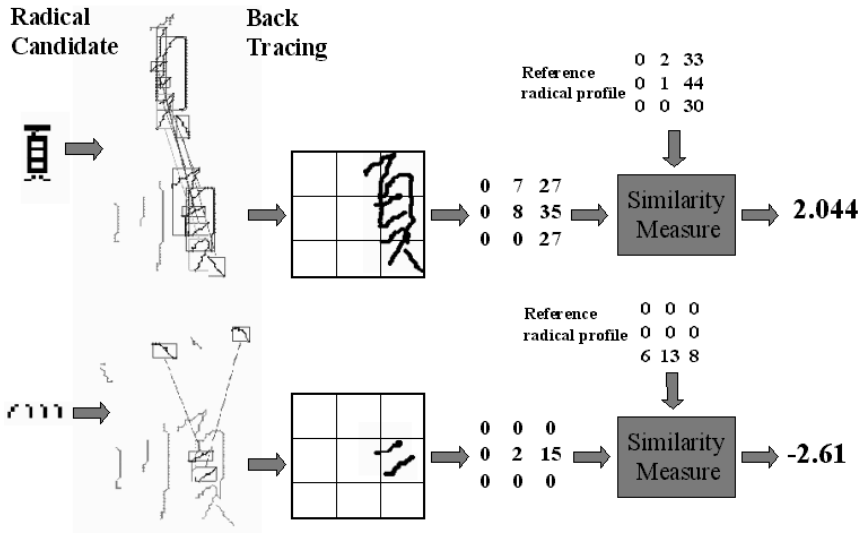

Figure 11: An example of radical detection, radical layout profile and similarity measure.

4.3 Classification

We use the results from radical extraction to obtain the most likely candidates thus reduce the range of searching. First of all, each character is represented by its extracted radical number X or NULL if there are no known radicals contained in the character, or X_1, X_2 if two radicals are extracted. Each handwritten annotation is then represented by a sequence of radical numbers. An example is illustrated in Figure 12, which is excerpted from the famous poem by Chinese famous poet Li Po. This famous poem has be used as the book cover of *Document Image Analysis* edited by Bunke, P.Wang and Bird.

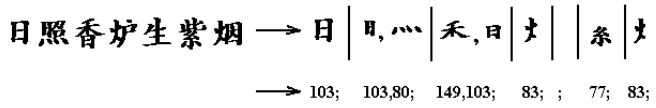

Figure 12: Radical representation of annotations.

The size of reference data can be reduced via a classification procedure in order to identify the most likely candidates. Each reference annotation is coded with a sequence of radical numbers by using radical extraction. When each query comes into the system, it is also converted to a radical code sequence via radical extraction. Edit distances are then computed on sequence of radical codes between the query and each item in the reference database using dynamic programming. The items in the reference database with least distances can be selected as most likely candidates. The time for computing edit distance is significantly less than that required for matching ink strokes ($1:40$) since the selection of most likely candidates is working on the sequence of extracted radical codes – a significant reduction of information from raw ink strokes. The number of most likely candidates to be

selected is a system parameter, it will be discussed more in Section 5.

Similar to the dynamic programming procedure in AIM, we have three basic operations in computing edit distances for radical codes: insertion, deletion and substitution. Instead of work on strokes, we now work on characters. Currently, the cost of insertion and deletion were assigned a constant number. The cost of substitution is determined by the number of occurrence that a radical code in one character does not exist in the other. For instance, if the two characters to be substituted are encoded as: $c_1 = 1, 3$ and $c_2 = 3, 17$, then the substitution cost is 2. If $c_1 = 1, 3$ and $c_2 = 14, 17$, then the substitution cost is 4. As can be seen, currently all the radicals are equally weight in computing the cost.

4.4 Matching

To make the matching more reliable, each annotation in the most likely candidate list will be matched against the query annotation using the same approximate ink matching as described earlier in this paper, and a top match list is generated. As illustrated in Figure 9, the matching can be applied to both ink vs. ink query and ink vs. typed text query, as the feasibility of matching typed text is still under exploration.

5 Experimental Evaluations

An experimental procedure was developed to systematically evaluate the components of semantic matching network. The entire semantic matching scheme is still under construction since it is comparatively new. However, the experimental results that will be described in this section show more than encouraging results.

In our experimental procedure, several issues have to be addressed and performance (feasibility) has to be evaluated. They are:

- Approximate ink matching - AIM can be used stand alone in our ink annotation application. It is also a core technique that is used by the radical extraction.
- Radical extraction - The key component in the semantic matching network.
- Classification - The classification for selecting most likely candidates has to allow false/missed radical extraction.

5.1 Data Collection

For our experiments, we collected 200 Chinese annotations, they are movie titles that are translated into Chinese. A graphical user interface (shown in Figure 13) is built to make the collection of handwriting easier. The content of the entry is displayed to prompt user to write with a stylus in the white rectangle area. Clicking "Next" button will automatically save the annotation and prompt the user for the next entry.

Four subjects were asked to write 200 Chinese annotations twice. They first wrote 200 annotations and these data were stored as reference ink database. Then they wrote the same 200 annotations again after a period of time, these data were stored as ink queries.

Figure 13: An ink collection tool.

5.2 Evaluation of Approximate Ink Matching

In our experiment, we allow each of the ink queries to match against all 200 annotations in the reference ink database. Since the contents of these two sets of data are known, we would know in advance what results we should expect. Each matching will yield a list of 200 values, each representing the distance between one item in the reference ink database and the query. These distances are then sorted and a rank number is given to each item.

When it comes to evaluating the retrieving performance of AIM, as does for any IR system, visualization can also play an important role in improving the system interactive recall/precision as stated by Veerasamy *et al* [21]. The top match list as shown in Figure 4 in our ink annotation system effectively displays the relevance judgement to the user without any burden on the processing time. In evaluating the AIM performance, we will observe how the retrieval rate changes with the number of top matches returned.

We have experimented with both VQ ink matching algorithm and elastic ink matching algorithm on the four data sets we collected. Figure 14 and Figure 15 show the recall for each individual writer using VQ and elastic ink matching versions respectively. The entire charts show recall as a function of the number of top matches returned. Although our experiments are run on a small test database, it is a promising result that for VQ ink matching algorithm, about 62% recall can be achieved at first hit. This number will be 80% if 5 top matches are viewed, that accounts only for 2.5% of the entire database. The VQ ink matching runs at about 2 seconds per query on a DEC Alpha workstation for our database. The performance has improved significantly for elastic algorithm. About 92% recall is achieved at first hit, this number will increase to 96% with 5 top matches. The speed for elastic ink matching is 160 seconds per query. As can be seen, the elastic ink matching algorithm is more reliable than VQ algorithm, yet the computation time is high.

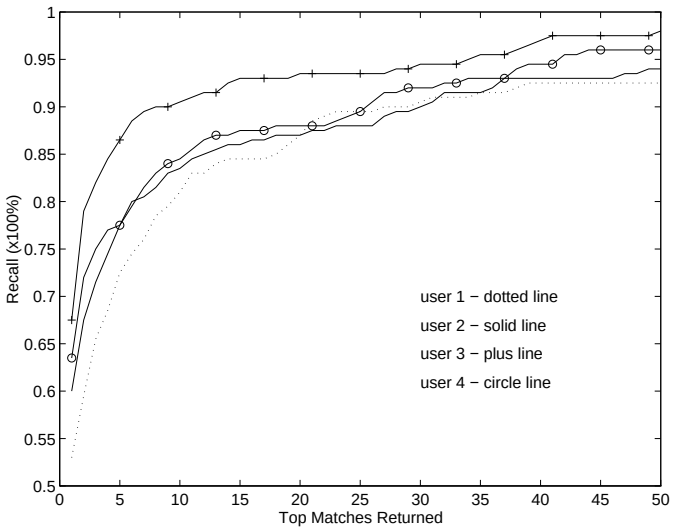

Figure 14: VQ matching: Recall vs. number of top matches returned for four users (represented by solid, "+" and "o" and "*" lines respectively).

5.3 Evaluation of Radical Extraction

The radical extraction algorithm uses word spotting elastic matching combined with layout detection. To evaluate this, the data is obtained from running on a small radical database which consists of 45 most frequently used radicals as identified by Xiao and Dai [24]. The radical extraction then runs on each item in our pre-selected ink database (consisting of characters that contain these frequently used radicals) against each radical. The recalls of radical extraction with and without using layout detection are illustrated in Figure 16. As can be seen, the performance of radical extraction significantly improves once radical layout is being used. Although we are still in the process of improving radical extraction and developing more robust and sophisticated algorithm, this result looks promising.

5.4 Evaluation of Classification

In classification, each character is converted to a radical representation or NULL if no known radicals is contained in the character. This is an information reduction at a rate of 3000 (characters) to nearly 200 (common radicals) or even less. An evaluation procedure was developed in order to test the feasibility of classification that relies on such information reduction.

In our experiments, we extract the radicals manually from 200 annotations in our previously collected database (translated movie titles in Chinese) and construct a sequence of radical code for each item. In this encoding process, we used the nearly 200 radicals listed in the radical index table in the Chinese-English Dictionary[5] and 45 mostly frequently used radicals identified by Xiao and Dai[24] respectively. Again, from 0 to 2 radical numbers are coded for each character. English letters, punctuations are ignored and simply coded

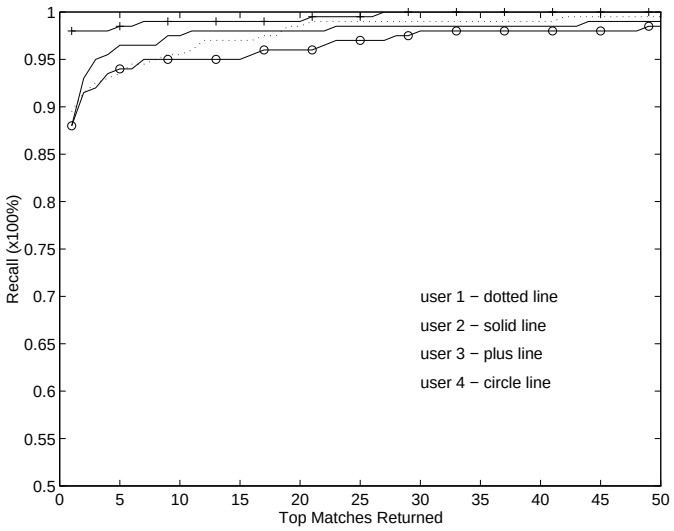

Figure 15: Elastic matching: Recall vs. number of top matches returned for four users (represented by solid, "+" and "o" and "*" lines respectively).

NULL. These manually extracted radical sequences only simulate the ideal case when all radicals are extracted perfectly. However, as will be described later in this section, we have to allow false or missed extraction of radicals.

In experiments, each radical coded annotation is compared with all other items in the entire database and a list of edit distances are computed using dynamic programming. By scanning each annotation at its edit distances, we obtain the number of hits at each distance from 0 to maximum distance. Figure 17 shows the average number of hits a query can get within a specific distance from 0 to maximum.

In Figure 17, we can interpret the distance axis as the margin for missing of radicals in radical extraction. As missed/incorrect radical rate increases, the distance increases, thus a higher number of hits occurs, which means there are greater number of most likely candidates within the database. The saving of computation time by classification depends on the number of most likely candidates that have to be selected for subsequent ink matching. At a given edit distance, a lower value in number of hits means more discrimination power for the classifier, thus the fewer number of most likely candidates can be selected for subsequent ink matching which in turn yields more saving of computation.

Furthermore, the saving of computation time for semantic matching is upper bound (minimum saving) by the horizontal line in Figure 17[1]. The curve below the line indicates the saving. The encouraging finding from this experiment is that under the upper bound line of saving, the classifier does have tolerance of distance which allows for miss/incorrect

[1]The timing is computed based on the fact that each query takes 160 seconds to match the entire database (200 titles), the radical extraction takes about 30 seconds for an average 5 character long annotation, and the classification time (computing edit distance for radical code sequences) is approximately 4 seconds per query.

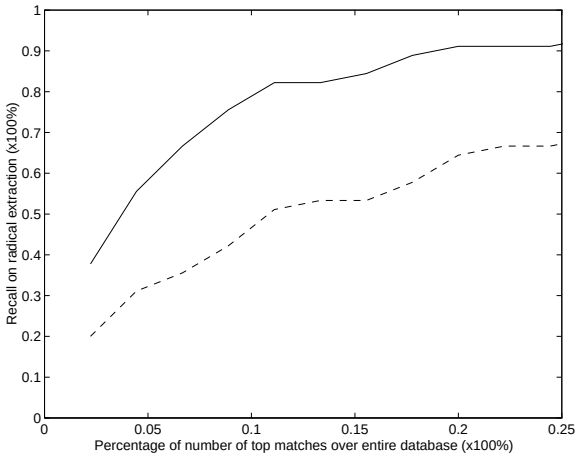

Figure 16: Recall on radical extraction with (solid line) and without (dashed line) radical layout detection.

radical extraction.

Also shown in Figure 17 is that the overall performance gain from semantic matching depends on the distance threshold, which in turn is determined by the performance of radical extraction. When less number of radicals are used for character encoding (e.g. 45 suggested by Xiao and Dai[24]), the saving of computation time for final ink matching is not as good as using 200 radicals, however, this makes radical extraction less time consuming and more reliable. The obtain of optimal number of radicals, the selection of representative radicals and the distance threshold on the selection of most likely candidates (classification) is a challenge task for the training process.

6 Conclusions

In this paper, we have described ink matching techniques that can be used for Chinese annotation search in a document system. Our ink annotation system allows a user to write notes anywhere on a page, and then later search for them as an aid to information retrieval. We have also given some experimental results on running ink matching algorithm on Chinese annotations. We have also performed systematic evaluations on several key components of our current system which are still under further development. The experimental results show that our basic approximate ink matching perform well on cursive Chinese handdrawn annotations with 92% recall for the first hit, and 96% recall for the top five matches returned. The experimental evaluations further show some promising results on radical extraction and classification along our line of semantic matching.

However, we think there are several ways in which the techniques we have described can be improved. Although the current semantic matching network uses semantics of Chinese for matching cursive handdrawn annotations, the fundamental matching technique of our system – approximate ink matching (AIM) – was developed previously for a language inde-

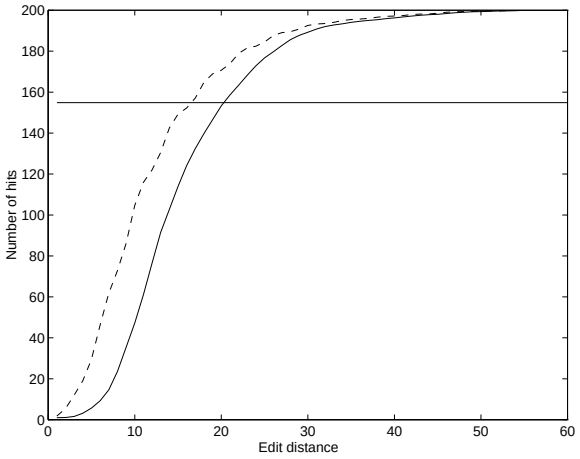

Figure 17: Average number of hits per query vs. distance. Solid line represents near 200 radical indice being used for encoding a character, dashed line represents 45 radical indice being used.

pendent environment. Therefore, a more complicated training procedure can be developed toward Chinese language and system parameters can be adapted to work better for Chinese handwriting. For instance, strokes are currently segmented at local minima of the y values. This is generally good for a system with mixed writing in Chinese and English. However, our informal test show that the choice of segmentation algorithm may have a significant impact on Chinese and English ink matching.

Further improvement can be explored in the selection of features in the VQ version of the algorithm. The feature set we use is generally good for common gestures, it may not be the best one for Chinese. This seems likely since Chinese is stroke based, the angles of strokes are particularly important. Changing the slope of a single stroke in a character slightly can change the meaning of the word. Other system parameters such as the weights for VQ features, the scaling factors for the basic costs in dynamic programming procedure may also be fine tuned through an empirical training process.

Finally, as stated in this paper, we are also doing further research on semantic matching. The extraction of radicals does not have to be perfect as illustrated in Section 4. However, the reliability of the radical extraction does affect the performance of the entire system. The number of radicals and the selection of radicals may be obtained from a training process as this is a component of *Learning by knowledge*. Some radicals are more immune to noise, while others are not. Careful selection of radicals to be used can increase the performance of radical extraction and thus the overall matching.

The ink matching scheme we proposed is more sophisticated than the conventional "syntactic" methods without contextual information [23][11]; it requires more memory and a backtracking procedure. More work need to be done in the future to overcome these difficulties. A larger dictionary (lexicon) and more intelligent machine of inferring and reasoning will also be helpful.

Acknowledgment

The authors of this paper would like to thank Andrew Tomkins for his early work of approximate ink matching and staff at the Panasonic Information and Networking Technologies Laboratory for their generous help on experimental evaluations.

References

[1] D.P. Ausubel, J.D. Novak, and H. Hanesian. *Educational Psychology: A cognitive view.* New York: HRW Co., 2nd Ed., 1978.

[2] F. Cheng and et al. Research on Chinese OCR in Taiwan. *Int. J. of Pattern Recognition and Artificial Intelligence*, 5(1,2), 1991.

[3] Sheng-Lin Chou and Wen-Hsiang Tsai. Recognizing handwritten Chinese characters by stroke-segment matching using an iteration scheme. In *Character and Handwriting Recognition. P.S.P. Wang (ed.)*, pages 175–197. World Scientific, 1991.

[4] Da Deng, K.P. Chan, and Yinglin Yu. Handwritten Chinese character recognition using spatial gabor filters and self-organizing feature maps. In *Proceedings of ICIP*, volume III, pages 940–944, Austin, Texas, November 1994.

[5] Beijing Institute of Foreign Languages Department of English. *A Chinese-English Dictionary.* Beijing Business Publishing House, 1985.

[6] J. Liu, W.K. Cham, and M.M.Y. Chang. Stroke order and stroke number free on-line Chinese character recognition using attributed relational graph matching. In *Proc. 13th ICPR*, pages 259–263, August 1996.

[7] Daniel Lopresti, Matthew Ma, and Jiangying Zhou. Approximate ink matching of Chinese handwritten annotations. In *Proceedings of the Seventeenth International Conference on Computer Processing of Oriental Languages*, pages 95–100, Hong Kong, April 1997.

[8] Daniel Lopresti and Andrew Tomkins. On the searchability of electronic ink. In *Proceedings of the 4th International Workshop on Frontiers in Handwriting Recognition*, pages 156–165, 1994.

[9] Matthew Ma, Patrick Wang, Daniel Lopresti, and Jill Crisman. Semantic matching of free-format Chinese handwriting. In *Proceedings of the Seventeenth International Conference on Computer Processing of Oriental Languages*, pages 107–111, Hong Kong, April 1997.

[10] R. Manmatha, Chengfeng Han, E.M. Riseman, and W.B. Croft. Indexing handwriting using word matching. *Proc. of the first ACM Intl. Conf. on Digital Libraries*, 1996.

[11] R.B. Millward. Models of concept formation. In *Aptitude, Learning, and Instruction. R.E. Snow et al (eds.)*. L. Erlbaum Assoc. Hillsdale, NJ, 1980.

[12] L. Mui, A. Agarwal, A. Gupta, and P.S.P. Wang. An adaptive modular neural network with application to unconstrained character recognition. In *Document Image Analysis. Bunke, Wang and Baird (eds.)*, pages 193–208. World Scientific, 1994.

[13] Hiroshi Murase. On-line recognition system for free-format handwritten Japanese characters. In *Character and Handwriting Recognition. P.S.P. Wang (ed.)*, pages 207–220. World Scientific, 1991.

[14] Masaki Nakagawa, Kimiyoshi Machii, Noaki Kato, and Toshio Souya. Lazy recognition as a principle of pen interfaces. In *INTERCHI93 Adjunct Proceedings*, pages 89–90, 1993.

[15] Ioannis Pavlidis, Rahul Singh, and Nikolaos P. Papanikolopoulos. An on-line handwritten note recognition method using shape. In *Proceedings of the International Conf. on Document Analysis and Recognition*, pages 914–918, 1997.

[16] Alex Poon, Karon Weber, and Todd Cass. Scribbler: A tool for searching digital ink. In *Companion Proceedings of the CHI*, pages 252–253, 1995.

[17] Dave Reynolds, Dipankar Gupta, and Richard Hull. Architectures for efficient scribble matching. In *Proceedings of the 4th International Workshop on Frontiers in Handwriting Recognition*, pages 488–495, 1994.

[18] Dean Rubine. Specifying gestures by example. *Computer Graphics*, 25, No. 4:329–337, 1991.

[19] Ju-Wei Tai. Some research achievements on Chinese character recognition in China. In *Character and Handwriting Recognition. P.S.P. Wang (ed.)*, pages 199–206. World Scientific, 1991.

[20] C.C. Tappert. Speed, accuracy, and flexibility trade-offs in on-line character recognition. In *Character and Handwriting Recognition. P.S.P. Wang (ed.)*, pages 79–96. World Scientific, 1991.

[21] Aravindan Veerasamy and Russell Heikes. Effectiveness of a graphical display of retrieval results. In *Proceedings of the 20th ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 236–245, 1997.

[22] Robert A. Wagner and Michael J. Fischer. The string-to-string correction problem. *Journal of the Association for Computing Machinery*, 21(1):168–173, 1974.

[23] P.S.P. Wang. Learning, representation, understanding and recognition of words - an intelligent approach. In *Fundamentals in Handwriting Recognition. S. Impedovo (ed.)*. Springer-Verlag, 1994.

[24] X.-H. Xiao and R.-W Dai. On-line handwritten Chinese characters recognition directed by components with dynamic templates. In *Proceedings of the Seventeenth International Conference on Computer Processing of Oriental Languages*, pages 89–94, Hong Kong, April 1997.

[25] S.L. Xie and M. Suk. On machine recognition of hand-printed Chinese characters by feature relaxation. *Pattern Recognition*, 21(1), 1988.

[26] K. Yamamoto and A. Rosenfeld. Recognition of handprinted Kanji characters by relaxation method. In *Proc. 6th ICPR*, pages 395–398, Munich, 1982.

[27] K. Yamamoto and H. Yamada. Recognition of handprinted Chinese characters and Japanese cursive syllabary. In *Proc. 7th ICPR*, pages 385–388, Montreal, 1984.