Combining High-Level Features with Sequential Local Features for On-Line Handwriting Recognition

Jianying Hu[1], Amy S. Rosenthal[2] and Michael K. Brown[1]

[1] Bell Laboratories, Lucent Technologies, 700 Mountain Avenue, Murray Hill, New Jersey 07974, USA.
{*jianhu, mkb*}*@research.bell-labs.com*
[2] Alpha Technologies Inc., 88 Centennial Avenue, Piscataway, New Jersey 08854, USA.
arosenthal@alpha88.com

Abstract. The trade-off between high-level, long-range features and low level, local features is common among many pattern recognition problems: the former are usually more powerful but less robust, while the latter is less informative but more reliable. In this paper we describe a new method for combining high-level long-range features and local features for on-line handwriting recognition. First, high-level features such as crossings, loops and cusps are extracted. A *localization* procedure is then applied to *spread* these high-level features over the neighboring sample points, resulting in local representations of nearby high-level features. These features are then combined with the usual local features at each sample point and used in an integrated segmentation and recognition process. This method allows incorporation of information carried by high-level long-range features while at the same time maintains the high reliability of the recognition system. We report experimental results on an HMM based recognizer for writer independent recognition of unconstrained handwritten words.

1 Introduction

The trade-off between high-level, long-range features and low level, local features is common among many pattern recognition problems: the former are usually more powerful but less robust, while the latter is less informative but more reliable. In on-line handwriting recognition, some attempts have been made on approaches based on high level features such as loops, crossings and cusps [1, 2], where each handwritten word is converted into a sequence of primitives based on extraction of high-level features. These methods do not work well on unconstrained handwriting, because the results of high-level feature extraction tend to be highly erroneous due to the large shape variations in natural cursive handwriting, especially among different writers. More recently, many powerful systems have been developed using simple local features such as slope angle and curvature, sampled at equi-arc-length points along an input script [3, 4, 5, 6, 7]. In such approaches, each sample point along the script is uniformly represented

by a local feature vector. The sequence of time ordered feature vectors is then processed through a network of statistical models (HMM's [3, 4, 5] or TDNN [6, 7]) and dynamic programming techniques are applied to find the best alignment between the input sequence and the models, thus providing the recognition result. These approaches, called *point oriented* methods, are much more robust because no premature shape classification is attempted before the recognition of the whole word; in other words, segmentation and recognition are integrated into a single process. On the other hand, they suffer from the loss of information carried by high-level features. While features formed by temporally nearby points such as cusps are represented implicitly to a certain extent through the slope angle feature, those formed by points that are spatially or temporally far apart, such as crossings and loops, are not presented to the recognition system at all. It is clear that to further improve such systems, high-level long-range features need to be incorporated.

Several approaches have been proposed before to incorporate certain long-range features into a point oriented system. Manke et. al. [8] proposed using local bitmap images called context bitmaps to model temporally long range but spatially short range phenomena such as crossings. However, this method can not model features such as loops, which are long-range both spatially and temporally. Hu et. al. [9] introduced interleaved segmental matching method to incorporate letter level shape matching into an HMM recognizer and reported substantial performance improvement. However, the letter shape feature used is still confined to a continuous temporal range and thus can not model events such as a crossing introduced by a delayed stroke in letter "t".

We propose a new method for combining high-level long-range features and local features. First, high-level features such as crossings, loops and cusps are extracted. Then, a *localization* procedure is applied to *spread* these high-level features over the neighboring sample points, resulting in local representations of nearby high-level features. These features are then combined with the usual local features at each sample point. Through this combination of high-level and local features, more information is represented in the features while the robustness of the system is still preserved. We report experimental results using the combined features on an HMM based recognizer [10] for writer independent recognition of unconstrained handwritten words.

The rest of the paper is organized as following. Section 2 briefly summarizes the preprocessing step. Section 3 discusses the method for combining local and high-level features. Section 4 describes the use of the new combined features in an HMM based recognizer. Section 5 presents experimental results on writer independent recognition of unconstrained handwritten words. We conclude in Section 6.

2 Preprocessing

Preprocessing is performed to improve the robustness of features to be extracted. It is composed of two parts: noise reduction and normalization. During noise

reduction, a smoothing spline filter is applied to the input script to reduce the effect of input device noise [3]. Cusps (points of abrupt directional change) are detected beforehand and treated as boundary points during smoothing, and thus preserved.

Normalization is then applied to reduce geometric variances in the handwriting data that are typically due to writing style differences. The size and orientation normalization is carried out using an algorithm based on the Hough transform [11]. First, local maximum and minimum points along the y coordinate are extracted. A modified Hough transform is then applied to extract parallel lines corresponding to the boundary lines separating ascender region, core body and descender region of the input word. This is followed by parameter refinement using linear regression. Each input handwritten word is then normalized to the horizontal orientation with a predefined core height (the height of a letter without ascender or descender, i.e., "a"). Then, a deskewing process is applied so that the dominant downward strokes are corrected to near-vertical. First, points of maximum and minimum y coordinates are identified and downward strokes (strokes going from a maximum point to a minimum point and longer than a threshold) are isolated. The straight segment of each downward stroke is then obtained by truncating the stroke from both ends until the angles formed by the two halves of the stroke differ by less than 10 degrees. The estimated skew angle θ is the average of the skew angles of all the straight pieces, each weighted by the length of the piece. After the skew angle estimation, each point in the script is then corrected by replacing x by $x' = x - y\tan(\theta)$.

The final step of normalization is re-sampling. In order to compensate for the variation in writing speed and device sampling rate, the handwritten word is re-sampled at equi-arc-length points.

3 Combining Local and High-Level Features

After preprocessing, a set of local features are computed at each sample point. These local features include the y coordinate, the tangent slope angle, and two invariant features called *normalized curvature* and *ratio of tangents*. The first two local features are commonly used in many on-line handwriting recognition systems. The latter two are more complex features we have developed that are invariant under translation, rotation and scaling [12, 10]. The incorporation of these two features which are independent of the size, location and orientation of the input word greatly improves the robustness of the system in dealing with unconstrained handwriting, where normalization does not always yield desired results.

Three high-level features are then extracted from the script: cusps, crossings, and loops. A cusp is defined as a point of abrupt change of drawing direction. Since the smoothing process in preprocessing has already eliminated the spurious wiggles while preserving the dominant direction changes [3], cusp extraction is performed simply by examining the angle formed by the line segments connecting three consecutive sample points. If this angle is smaller than a preset threshold,

the center point is labeled as a cusp. Figure 1 shows a sample word with the cusps highlighted.

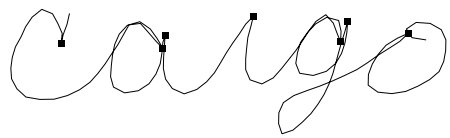

Fig. 1. Sample word with cusps indicated.

A crossing occurs when two *separated* segments intersect. Two segments along a script are considered separated if the distance between them along the drawing trajectory is larger than a threshold. When such an intersection is detected, the end point closer to the intersection on each intersecting segment is labeled a crossing point. To make the feature more robust, two segments are considered intersecting if the distance between one pair of end points is smaller than a preset threshold. To find all crossings, an input script containing n points is sequentially searched for pairs of end points of separated segments which are located nearest each other in the x-y space. Two points within a selected maximum radius of each other are initially labeled crossings. Then, a refinement procedure is applied to neighboring points to determine if there is a pair of points which lie even closer together than the initial pair. The crossings are then relabeled if such a pair is found. By this method, each loop is identified by a unique pair of sample points. Figure 2 shows a sample word with the crossing points highlighted. For clarity only the first one of the labeled pair of points is highlighted for each crossing. Notice that the intersections caused by small loops in the letters "h" (lower-left corner) and "n" (upper-left corner) are not classified as crossings because they do not satisfy the distance threshold for separated segments. The rational is that these intersections tend to be incidental and highly dependent on each particular writer's writing habit, and therefore should be ignored.

Fig. 2. Sample word with crossing points indicated.

The segment between two crossing points is declared a loop if there is no pen-up in between. Figure 3 shows the same sample word as shown in figure 2 with the loops highlighted.

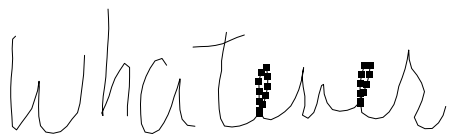

Fig. 3. Sample word with loops indicated.

Once the high-level features have been detected, the next question is how to represent them in the feature vector assigned to each sample point. The loop feature is relatively straightforward – we can simply assign a binary value to each sample point indicating whether or not the point is on a loop. However, the cusp and crossing features are not so trivial. If we apply the same scheme and define a feature with value 1 at a cusp (crossing) and 0 at all other points, the corresponding feature sequence will be composed of consecutive 0's with occasional occurrences of single 1's. These occasional 1's will most likely not make any difference in recognition because they are statistically insignificant. We shall call features like cusps and crossings *point-centered high-level features*. The above analysis shows that the effective use of such point-centered high-level features requires attaching more importance to certain points than others. This requirement can not be satisfied directly by the basic framework used for local features, where each sample point is represented uniformly.

Our solution to this problem is to "spread" the influence of each point-centered high-level feature to its neighboring points. Instead of using the feature itself, we use a derived feature which measures the distance along the script from each sample point to the nearest cusp or crossing point. The value of this derived feature is 0 at the labeled high-level feature point and positive for all nearby points, where a larger value indicates that the point is further away from the nearest labeled sample point. If at any point there is no labeled high-level feature point within a predefined maximum distance w, then the value w is assigned for that point. The two features derived using this scheme are called *cusp distance* and *crossing distance*, respectively. Notice that this is only one of the many possible ways to "spread" the influence of a point-centered high-level feature. For example, another, perhaps more intuitive, approach is to apply a kernel (e.g. Gaussian) centered at each feature point. In this case the labeled feature point will assume a predefined maximum value a, while nearby points will have decreasing values as they become further away from the labeled feature point. We chose the first approach because it requires less computation and appears to have the same effect as the second approach. The idea of spreading high-level

features seems to be intuitive in human recognition of handwriting as well: the occurrence of high-level features does not only affect the perception of the exact point, but also the regions surrounding it.

As mentioned before, the high level features themselves are not robust features for unconstrained handwritten words, as can be seen from the samples shown above. However, we believe that they do contain valuable information which, properly incorporated, can improve the recognition system's performance. The three high level features described above are combined with the local features mentioned at the beginning of this section, forming a 7 dimensional feature vector at each sample point. These 7 features are: tangent slope angle, y coordinate, normalized curvature, ratio of tangents, crossing distance, cusp distance and loop.

4 Using the Combined Features in An HMM Recognizer

The combined features are used in an HMM based recognizer called AEGIS [10], designed for writer-independent recognition of unconstrained handwritten words.

The system is composed of *nebulous* stroke models embedded in an evolutional grammar network representing the vocabulary [10]. Each arc in the grammar network corresponds to a letter model representing a unique letter pattern class. Each letter model is a left-to-right HMM with no state skipping (see figure 4), which is composed of a sequence of a variable number of stroke models as specified in a lexicon, and each stroke model is a single state HMM. A letter typically requires more than one letter model representing different letter pattern classes resulted from different writing styles. Delayed strokes are considered special one-stroke letters. A word with delayed strokes is represented in the grammar network by a number of alternative paths corresponding to all possible rendering sequences with delayed strokes occurring in different positions. Ligatures are also modeled by single-state HMM's which are inserted between adjacent characters, but with the option of skipping. The HMM's are trained using the standard iterative segmental method [13]. During training, a separate probability distribution is estimated for each feature at each model state. At recognition time, a combined log-likelihood score of a particular feature vector being generated by a certain state is calculated as the weighted sum of the individual feature log-likelihood scores [12].

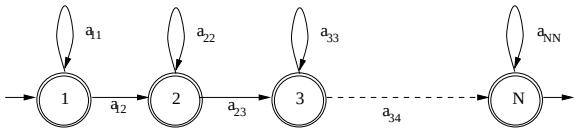

Fig. 4. A left-to-right HMM with no state skipping.

During recognition, the evolutional grammar network is expanded and the embedded HMM's are instantiated at run time [10]. Each arc is then replaced by the HMM referred to by the corresponding label. Segmentation and recognition are integrated into one decoding process using the Viterbi algorithm. For large vocabulary applications, recognition is carried out in two steps: constrained N-best decoding [14] to quickly find the top N most likely candidates, followed by re-scoring using interleaved segmental matching where whole letter shape matching scores are incorporated [9].

5 Experimental Results

Experiments were carried out using unconstrained handwritten lower-case word samples from 66 writers. The samples were collected at Bell Laboratories at Holmdel and Murray Hill. The words were randomly drawn from a 25,000 word English dictionary.

For this experiment, the HMM models were trained using a total of about 6000 samples from 50 writers. The test set contains 125 samples from each of the other 16 writers, totaling 2000 samples. Since words were randomly selected during sample collection, there is little correlation between the vocabulary used in training and the one in recognition. A lexicon of 1905 words, covering all unique words in the test set, was applied in recognition. Table 1 shows the error rate of the system when only the four local features are used, compared to the error rate when the three high-level features are added. As can be seen from the table, the incorporation of high-level features has reduced the error rate by about 14%.

Table 1. Writer independent recognition results with different features.

Feature Set	4 local features only	+ 3 high-level features
Error Rate	10.4%	9.0 %

6 Conclusion

We have described a new method to combine high-level long-range features with local features for on-line handwriting recognition. The method allows incorporation of information carried by high level features such as cusps, crossings and loops while at the same time maintains the high reliability of the recognition system. A localization process is applied to the high-level features to yield local representations of these features which can be readily incorporated into an integrated segmentation and recognition process. The method has been tested

on an HMM based recognizer for a task of writer-independent recognition of unconstrained handwritten words. Our experiments show that the incorporation of the high-level features results in about 14% error rate reduction compared to using local features alone.

References

1. S. A. Guberman and V. V. Rozentsveig. Algorithm for the recognition of handwritten text. *Automation and Remote Control*, 37 37(5):751–757, May 1976.
2. S. Bercu and G. Lorette. On-line handwritten word recognition: An approach based on hidden markov models. In *Proceeding Third Int. Workshop on Frontiers in Handwriting Recognition*, pages 385–390, Buffalo, USA, May 1993.
3. J. Hu, M. K. Brown, and W. Turin. Handwriting recognition with hidden Markov models and grammatical constraints. In *Proc. 4th IWFHR*, pages 195–205, Taipei, Taiwan, December 1994.
4. J. Makhoul, T. Starner, R. Schartz, and G. Chou. On-line cursive handwriting recognition using speech recognition methods. In *Proc. IEEE ICASSP'94*, pages v125–v128, Adelaide, Australia, April 1994.
5. K. S. Nathan, H. S. M. Beigi, J. Subrahmonia, G. J. Clary, and H. Maruyama. Real-time on-line unconstrained handwriting recognition using statistical methods. In *Proc. IEEE ICASSP'95*, pages 2619–2622, Detroit, USA, June 1995.
6. S. Manke and U. Bodenhausen. Npen++: A writer independent, large vocabulary on-line cursive handwriting recognition system. In *Prod. 3rd ICDAR*, pages 403–408, Montreal, Canada, August 1995.
7. M. Schenkel, I. Guyon, and D. Henderson. On-line cursive script recognition using time delay neural networks and hidden Markov models. *Machine Vision and Applications, Special Issue on Cursive Script Recognition*, 8, 1995.
8. S. Manke, M. Finke, and A. Waibel. Combining bitmaps with dynamic writing information for on-line handwriting recognition. In *Prod. 12th ICPR*, pages 596–598, Jerusalem, October 1994.
9. J. Hu, M. K. Brown, and W. Turin. Use of segmental features in HMM based handwriting recognition. In *Proc. IEEE SMC'95*, pages 2778–2782, Vancouver, Canada, October 1995.
10. J. Hu, M. K. Brown, and W. Turin. HMM based on-line handwriting recognition. *IEEE PAMI*, 18(10):1039–1045, October 1996.
11. Amy S. Rosenthal, J. Hu, and M. K. Brown. Size and orientation normalization of on-line handwriting using Hough transform. In *Proc. ICASSP'97, to appear*, Munich, Germany, April 1997.
12. J. Hu, M. K. Brown, and W. Turin. Invariant features for HMM based handwriting recognition. In *Proc. ICIAP'95*, pages 588–593, Sanremo, Italy, September 1995.
13. L. R. Rabiner and B. H. Juang. *Fundamentals of Speech Recognition.* Prentice Hall, Englewood Cliffs, NJ, 1993.
14. J. Hu and M. K. Brown. On-line handwriting recognition with constrained n-best decoding. In *Proc. 13th ICPR*, volume C, pages 23–27, Vienna, Austria, August 1996.

This article was processed using the LaTeX macro package with LLNCS style