

Comparing Isolated Symbol Recognition Methods

Scott MacLean <smaclean@uwaterloo.ca>, George Labahn <glabahn@cs.uwaterloo.ca>



Symbolic Computation Group, University of Waterloo

Context

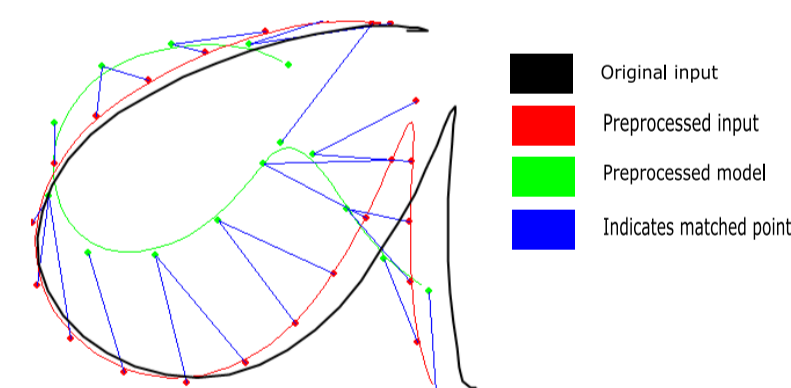
- Any application using a pen-based interface to capture information from the user (aside from the shape of the digital ink) must have a mechanism for determining what the user has written or drawn.
- We are specifically interested in recognizing the symbols that comprise mathematical expressions in order to extract mathematical meaning from the user's writing and to perform computations on it.

Common Elements

- A basic technique in symbol recognition is to record several samples of writing for each symbol of interest, then to compare the user's input against those stored models.
- Input and model symbols both consist of time-ordered sequences of strokes. Strokes are time-ordered sequences of (x,y) coordinates.
- Prior to any recognition, strokes are preprocessed by joining broken strokes, smoothing, subdividing, trimming jagged ends, and normalization (preserving aspect ratio).

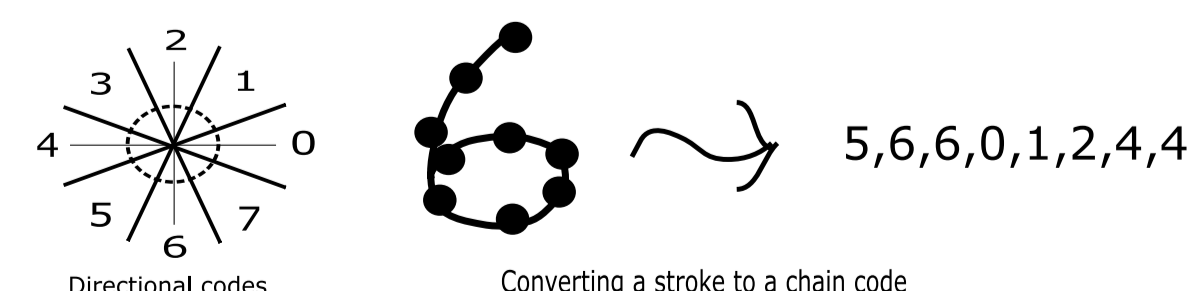
Elastic Matching

- A variant of "Dynamic Time Warping", commonly used in speech recognition. We treat the sequence of x-coordinates and the sequence of y-coordinates as two sampled signals, with a distance metric between model and input points that combines the signals.
- Each of the input points is matched to the "best" model point; however, the first and last points in the model must be matched to, and if the i th input point is matched to the j th model point, then the $i + 1$ st input point must be matched to the j th, $j + 1$ st, or $j + 2$ nd model point.
- A distance metric measures the cost of matching an input point to a model point. The algorithm finds the smallest possible total cost while ensuring every input point is matched.
- Our distance metric is Manhattan distance plus the difference in stroke tangent angle.
- Derived from [Tap82].



Structural Chain-codes

- Converts input and model strokes to sequences of digits called "chain-codes" based on stroke direction between adjacent points. Digit sequences are compared using string-matching techniques.
- Divide the $[0, 2\pi)$ interval into N subintervals. If the angle of the line segment between point i and $i + 1$ of a stroke falls in the k th subinterval, then the i th digit in that stroke's chain-code is k .
- Match score is calculated by treating chain-codes as vectors, taking the (modular) difference between input and model vectors, and applying some norm. (We use the 1-norm).
- The chain-codes are compared in a way that allows for slight rotation and truncation of strokes.
- Our implementation is based on [Cha00].



Stroke Feature Extraction

- Extracts information about each stroke from the input and model and compares the vectors of feature information. The match score is obtained by taking some norm of the vector difference.
- Information we extract includes: left- and top-most coordinate, width, height, first and last point coordinates, and total arclength.
- The norm can be used to weight the relative importance of each feature.

Direction Element Method

- A feature extraction method which captures stroke direction information.
- Divide the $[0, 2\pi)$ interval into N subintervals as in chain-code matching.
- For a given symbol, partition the bounding box into a grid and extract stroke angle information from each grid cell, yielding a feature vector.
- Each pair of adjacent points induces contributions to the two cells nearest to the points and the angle between them. These contributions are proportional to the distance between the points. The nearer that angle is to a subinterval boundary, the larger the influence on that subinterval's vector entry.
- The feature vector is the sum of all these contributions over all pairs of adjacent points. A norm is applied to the difference in model and input feature vectors to determine the match cost.
- From [Kan99].

Combining Recognizers

- Combining the results of several recognizers can give more accurate results than using only one.
- We use a voting method, similar to [Kan99]:
 - The candidate rankings of each recognizer are truncated (we use at most five candidates from each). The poorest candidate's score is used as a baseline.
 - Each candidate's relative score is given by the ratio of the baseline to its original score. (Lower scores indicate better matches until this step is completed).
 - A weighted sum of these relative scores over all recognizers gives the final score. Normalization yields percentage confidences.
- This method, since it uses score ratios, is insensitive to the different score ranges produced by various recognition methods.

Database Pruning

- There are currently 103 symbol types in our database, with many samples for each. To compare input against every sample using several recognizers would be very time consuming.
- Some matchers are much faster than others.
- We therefore use the fast matchers with loose thresholds to remove symbols from the search space if they clearly do not match.
- In practice, this leaves only a handful of symbols to run the slower, more precise matchers on. However, this pruning may remove the correct symbol from the search space.
- This approach is similar to the one taken in [Wat05].

Experimental Results

A1 and A2 each constitute half of an isolated character set by one writer. B is a collection of mathematical expressions samples by multiple writers. C is the training database used in practice with samples from multiple writers.

Results are presented as top5;top1.

Train Set	Test Set	No. Tests	EM	CC	SF	DE	EM+DE	CC+SF
C	B	691	648;581	649;542	596;400	634;508	648;579	648;574
	no pruning	(%)	93.8;84.1	93.9;78.4	86.3;57.9	91.8;73.5	93.8;83.8	93.8;83.1
A1	A2	625	685;605	628;299	606;396	581;379	685;578	649;366
	no pruning	(%)	99.1;87.6	90.1;43.3	87.7;57.3	84.1;54.8	99.1;83.6	93.9;53.0
A2	A1	625	596;555	595;522	588;451	592;521	596;553	596;530
	no pruning	(%)	95.3;88.8	95.2;83.5	94.1;72.2	94.7;83.4	95.4;88.4	95.4;84.8
A2	A1	625	625;577	593;421	599;442	567;414	625;566	509;468
	no pruning	(%)	100;92.3	94.9;67.4	95.8;70.7	90.7;66.2	100;90.6	97.4;74.9
A2	A1	625	614;579	614;554	603;468	614;539	614;577	613;551
	no pruning	(%)	98.2;92.6	98.2;88.6	96.5;74.9	98.2;86.2	98.2;92.3	98.1;88.2
A2	A1	625	624;581	591;438	606;454	586;461	624;579	608;474
	no pruning	(%)	99.8;93.0	94.6;70.1	97.0;72.6	93.8;73.8	99.8;92.6	97.3;75.8

- Most errors where the candidate was not in top1 but was in top5 are caused by confusion between, eg., 0/o/O, x/X, 1/(l)/j, 2/z/Z.
- In the tests where database pruning was used, the matchers often hit a ceiling on the number of correct results caused by the pruning algorithm removing the correct symbol from the search space. Clearly the pruning algorithm must be improved.
- For many matchers, pruning improves recognition accuracy.

Other Considerations

- In practice, input strokes must be segmented (ie. determine which strokes comprise a single input symbol). Segmentation was disabled for the tests described above.
- For symbols of more than one stroke, we take the average or maximum match score over all strokes (rather than a sum) to avoid a bias toward single-stroke symbols.
- Before input is compared to a model, strokes are joined together, reordered, and reversed as appropriate to best match the model symbol.

References

- [Cha00] Kam-Fai Chan and Dit-Yan Yeung. *Recognizing on-line handwritten alphanumeric characters through flexible structural matching*. Pattern Recognition, 32(7), pp. 1099-1114, 2000. Available at ftp://ftp.cs.ust.hk/pub/kchan/paper/chan.pr99.pdf
- [Kan99] T. Kanahori, K. Tabata, W. Cong, F. Tamari, M. Suzuki. *On-Line Recognition of Mathematical Expressions Using Automatic Rewriting Method*. Proceedings of the Fourth Asian Technology Conference on Mathematics, Guangzhou, pp. 291-300, 1999. Available at http://www.inftyproject.org/files/1999_ATCM.Okamura.pdf
- [Tap82] Tappert, C.C. *Cursive Script Recognition by Elastic Matching*. IBM Journal of Research & Development 26(6), pp.765-771, 1982.
- [Wat05] Stephen M. Watt and Xiaofang Xie. *Prototype Pruning by Feature Extraction for Handwritten Mathematical Symbol Recognition*. Maple Conference 2005 Proceedings, pp.423-437, 2005.