An incremental recognition method for online handwritten mathematical expressions

Conference Paper · November 2015

DOI: 10.1109/ACPR.2015.7486488

4 authors, including:

Cuong Tuan Nguyen
Vietnamese-German University
52 PUBLICATIONS **546** CITATIONS

SEE PROFILE

Anh Le Duc
The Institute of Statistical Mathematics
47 PUBLICATIONS **1,241** CITATIONS

SEE PROFILE

Masaki Nakagawa
Tokyo University of Agriculture and Technology
314 PUBLICATIONS **3,351** CITATIONS

SEE PROFILE

An Incremental Recognition Method for Online Handwritten Mathematical Expressions

Khanh Minh Phan, Cuong Tuan Nguyen, Anh Duc Le, Masaki Nakagawa
Department of Computer and Information Sciences
Tokyo University of Agriculture and Technology
{pmkhanh7890,ntcuong2103,leducanh841988}@gmail.com, nakagawa@cc.tuat.ac.jp

Abstract

This paper presents an incremental recognition method for online handwritten mathematical expressions (MEs), which is used for busy recognition interface (recognition while writing) without large waiting time. We employ local processing strategy and focus on recent strokes. For the latest stroke, we perform segmentation, recognition and update of Cocke-Younger-Kasami (CYK) table. We also reuse the segmentation and recognition candidates in the previous processes. Moreover, using multi-thread reduces the waiting time. Experiments on our data set show the effectiveness of the incremental method not only in small waiting time but also keeping almost the same recognition rate of the batch recognition method without significant decrease. We also propose the combination of the two methods which succeeds the advantages of the both.

1. Introduction

Mathematical Expressions (MEs) are very popular in education, science, engineering, business and even in daily life. There are basically three ways to input MEs to a computer. A user can input them in a math description language like LATEX or employ an editor like Microsoft Equation Editor (MEE). These two methods are the most popular methods, but the problem is that the user must remember the grammar of math symbols/expressions or select menus and find symbols/expressions in a long list. These methods are usable for scientists, engineers, teachers, businessmen and other professionals but awkward for ordinary people. The third method is to use a recognizer for handwritten MEs. A user just writes math symbols/expressions that he/she wants to input and the math recognizer will recognize and translate them to LATEX or MEE automatically.

To get high recognition rate, the best ways is to recognize the online handwritten ME after the whole ME is completed because all of context information is available. We call this as batch recognition [1]. This method is suitable for a user who does not need feedback from ME recognition while writing and they want to receive only a final result after breaking writing. Misrecognitions and even recognitions during writing may interrupt user's thinking. We call this user interface as lazy interface [2]. In these days, touch-based or pen-based devices using display integrated tablets are spreading very quickly into our daily life and their writing surfaces are getting larger. They can return the feedback of recognition in real-time and enable a user to interact with the feedback e.g. erase and rewrite strokes when a mis-recognition occurred. We can take an advantage from this technology by recognizing after every new stroke is written. We call this kind of recognition as busy recognition interface.

An incremental recognition can be applied for not only busy recognition interface but also the lazy recognition interface. Scott MacLean et al. show his approach for incrementally recognizing handwritten mathematics by using relational grammars and fuzzy sets [3]. In his method, an incremental parsing of two-dimensional input is presented.

When a new stroke is input, the segmentation is updated, this new stroke and its previous strokes are recognized, a new node is created which includes the information of the new stroke and its recognition, and then these nodes are used to update the information table. This process is repeated on recent strokes rather than on full ME so that ME recognition result is shown immediately after writing is finished without noticeable waiting time while keeping high recognition rate.

Here, we define some terminology. A stroke is a time-sequence of pen-tip or fingertip coordinates from pen (finger)-down to pen (finger)-up. A vector from pen (finger)-up to pen (finger)-down is called an off-stroke. A time-sequence of strokes and off-strokes is called digital ink.

Due to the development and wide spread of both the interactive and non-interactive devices, the environment is favorable for people to write MEs to input them into a computer.

The current dominant methods to input MEs using LATEX or MEE will be used for professionals, but we hope that expanding users and applications may need more casual methods and accept our method as one of them combined with our improved math recognizer.

The rest of the paper is organized as follows. Section 2 presents an overview of the batch recognition method. Section 3 describes the incremental method. Section 4 presents recognition experiments of the incremental recognition. Section 5 is our conclusion.

2. Overview of batch recognition

This section gives an overview of the batch recognition method. Firstly, the geometric features are calculated from all the input strokes. In the batch recognition, segmentation and recognition are applied for all the sequence of strokes based on these features. Finally, the Cocke-Younger-Kasami (CYK) [4] algorithm is used to analyze and find the best recognition of handwritten MEs.

2.1. Symbol segmentation

In this step, the separation probability of each pair of adjacent strokes is computed from 21 geometric features [5]. Then, an SVM classifier is applied for segmentation.

The symbol hypotheses are generated from the SVM output. Invalid hypotheses, which have more than 5 strokes are discarded, since the maximum number of strokes for mathematical symbols is 4 and a split of a stroke or 1 additional stroke is assumed.

2.2. Symbol recognition

Character recognizer is utilized to recognize each symbol hypothesis from the previous task. The combined recognizer composed of offline and online recognition methods [6] is a robust version because it has advantages of both the methods. Particularly, it could recognize connected strokes or cursive strokes by the online method and stroke disorders or duplicated strokes by the offline method.

Almost all the Japanese character recognizer is reused for mathematical symbol recognition with three main modifications. Firstly, all categories in Japanese dictionary are replaced by a hundred mathematical symbol categories. In addition, only three candidates, which have highest results in recognition, are remained for each symbol pattern. Finally, the process is extended to recognize 'period', 'comma' and 'prime'.

For each recognized symbol, the candidate of 'period'/ 'comma'/ 'prime' is added. If the symbol has both height and width less than a half of the average height (H) and width (W) of current ME, respectively, its probability is calculated as follows:

$$P_{period/comma/prime} = F(height) * G(width)$$

Where F and G are fuzzy functions as shown in Figure 1.

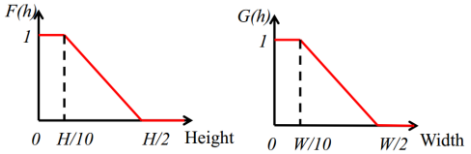

Figure 1: Membership function for "period/comma/prime"

2.3. Structural relation

Structures or relations among symbols in MEs are ambiguous even for human in some cases. Hence, a body box, which includes the main body of each symbol, is utilized to extract relations. All symbols are classified into four groups: ascendant, descendant, normal and big symbols. Ascendant symbols extend above the mean line while descendant symbols elongate below the base line. Normal symbols typically stay between mean line and base line. On the contrary, big symbols go over the base line and mean line. Figure 2 illustrates four groups and their positions with mean line and base line. For each group, a body box is defined. It should be noted that the body box for each symbol pattern is different for each symbol recognition candidate.

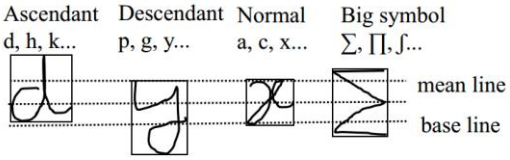

Figure 2: Body boxes

The feature Dx is extracted which shows the relation between the horizontal centers of 2 body boxes for sub-expressions, to divide above, below, inside relations into Group 1 and horizontal, superscript, subscript relations into Group 2. Then, the features H and Dy are utilized to classify the above, below, inside relations in Group 1 by a SVM while H, Dy and O features are used to classify the horizontal, superscript, subscript relations in Group 2 by another SVM. The calculation and illustration of these features are displayed in figure 3.

2.4. Structure analysis

This analysis solves all local ambiguities in previous phases. A 2D-SCFG is defined formally by a five-tuple G= (N, Σ, R, P, S) where:

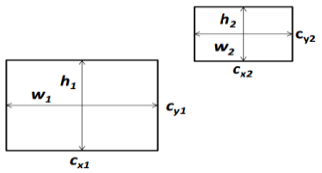

Figure 3: Feature of structural relations

- N is a finite set of non-terminal symbols.
- Σ is a finite set of terminal symbols.
- R is a finite set of relations between 2 sub-expressions. These relations are horizontal, over, under, superscript, subscript, inside. P is a finite set of productions of one of the following forms:
  - $X \rightarrow a$
  - $X \rightarrow A$
  - $X \rightarrow r\ A\ B$, with $X, A, B \in N$, $a \in \Sigma$, $r \in R$.
- $S \in N$ is a distinguished start symbol.

Although handwritten strokes may be ambiguous, a grammar is defined to be unambiguous. This implies that every valid grammar expression has a unique leftmost derivation. The unambiguous grammars are difficult to be defined in Chomsky Normal Form (CNF). Therefore, one more production rule ($X \rightarrow A$) is added for making grammar definition easier.

2.5. CYK algorithm and its implementation

In the first place, the system is configured by a definition of SCFG. Then, the configured system invokes the CYK algorithm to produce a list of candidates for each input sequence of strokes ($s = s_1, s_2, \ldots s_n$) expressing an ME. The algorithm has two steps as follows:

Initial step: The first five (the maximum number 4 for math symbols plus 1) rows in the CYK table are initialized.

Parsing step: CYK operates only on SCFGs given in CNF. Additionally, the original CYK is modified by adding more production rules ($X \rightarrow A$). $X \rightarrow r\ A\ B$ production rules are employed to reduce 2 sub-MEs to a non-terminal. $X \rightarrow A$ production rules are employed to reduce the non-terminal further to form another non-terminal. For an ME, 40 best candidates are saved at each cell in the CYK table. The candidates of the final result are extracted at cell (1, N).

In parsing step, stroke order is utilized to arrange strokes instead of traveling all strokes. On the other hand, all particular writing orders such as those before and after fractions, roots, and parentheses in the grammar are prepared to avoid parsing failures.

3. Incremental recognition method

This research not only aims to reduce waiting time as much as possible but also keeps the recognition rate as high as the batch style. In the batch method, symbols recognition and building the CYK table are time consuming. In this section, we introduce a method that is processed in the background. Furthermore, ME recognition is displayed without any noticeable delay.

3.1. Processing flow

Figure 4 demonstrates the processing flow of the incremental recognition method.

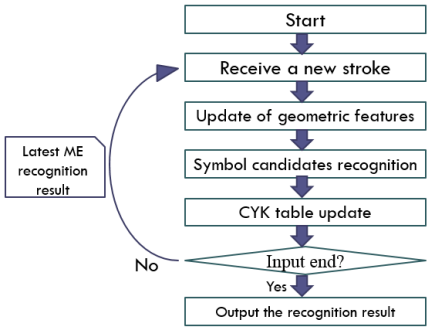

Figure 4: Flow of incremental recognition

To begin with, the system receives a new stroke from user's writing. In the second step, we update the geometric features and segmentation. In the third step, our method recognizes the symbols that related to the recently received stroke. After that, the CYK table is updated and we analyze the structure to get the ME recognition result. Finally, the latest result is reused for the next processing cycle.

3.2. Update of geometric features

In this step, 21 geometric features are calculated from the recently received stroke. Based on this new information, the average values of geometric features of the current ME are updated immediately.

3.3. Symbol candidates recognition

In the processing cycle, the result of the ME recognition is updated to the latest stroke. The maximum strokes that form a mathematical symbol is limited within four plus one strokes. Consequently, a recently received stroke affects up to four previous strokes. Namely, this new stroke may itself form a symbol or combine with up to four previously written strokes to build up a symbol. All the candidates that have been recognized before are considered stable. All the new candidates related to the new stroke are updated to the recognition table.

3.4. CYK table update

After all the candidates are recognized, they will be used as main components for the CYK algorithm. The CYK algorithm for incremental recognition is similar to the batch style that has two steps

The main idea of building the CYK table in batch style is the construction in ascending order vertically. It is compulsory to form the first row completely in advance of continuously adding nodes into cells of the upper rows. Under those circumstances, incremental recognition is an inapplicable method. On the contrary, in incremental recognition method, the latest stroke is independent to the CYK table which is built from the previous strokes.

Therefore, the CYK table only needs to be expanded without changing the existing table whenever it receives a new stroke. Specifically, after receiving a new stroke, a new cell at the first row is created, and then it will combine with other cells on the left side for generating cells in the upper rows.

Initial step: The maximum number of strokes per symbol is four plus 1. We initialize up to 5 rows in the CYK table for a newly received stroke as algorithm 1.

Algorithm 1
for $j = 0 .. 3$
  if $n > j$
    for $i = n\text{-}j\text{-}1 .. n\text{-}j$
    t = group of strokes $s_i \ldots s_{i+j}$
    if t does not satisfy constraints of rejecting invalid hypotheses
      for each production $X \rightarrow a$
        if $(P_{reco}(t|\ a)>0)$
          Add node $(X \rightarrow a, P_{rec}(t|\ a)* P_{seg},t)$ into cell (j,i)

Parsing step: X→ r A B production rules are employed to reduce 2 sub-MEs to a non-terminal. X → A production rules are employed to reduce the non-terminal further to form another non-terminal. For an ME, 40 best candidates are saved at each cell in the CYK table in the same way as the batch method. The CYK table is built, however, from left to right rather than from bottom to top. For each processing time, the candidates of the final result are always extracted from the cell (1, n) as algorithm 2.

Algorithm 2 (with $n >1$)
for $i = 1 .. n\text{-}1$
for $j = n – i - 1$
for $k = 0 .. i$
for each rule $(X \rightarrow r\ A\ B)$
  $C1 = getcell(k, j)$
  $C2 = getcell(i\text{-}k\text{-}1, j+k+2)$
  $prob = P(C1|A) * P(C2|B) * P_{rel}(C1,C2|r) * P_{seg}(s_{j+k},s_{j+k+1}) * P_{gram}(X \rightarrow r\ A\ B)$
  If $(prob>0)$ {
    Add node $(X \rightarrow r\ A\ B,prob,C1,C2)$ into cell (i,j)
  }
For each rule $(X \rightarrow A)$
  If cell(i,j) has node A {
    Add node $(X \rightarrow A, P_A)$ into cell(i,j)
  }

3.5. An example of the processes

Figure 5 presents an example of building the CYK table in the incremental recognition. In this example, a user intends to write “2+86”. The parenthesized number in the left denotes the step after every new stoke is input and the diagram in the right shows the CYK table after every input

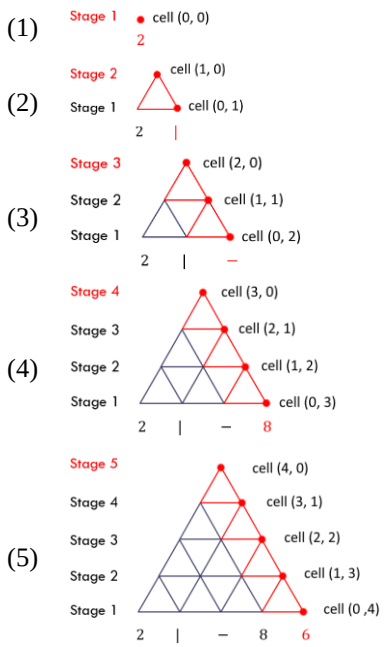

Figure 5: Building the CYK table for “2+86”.

stroke. The strokes at the bottom of each diagram are input strokes in writing order. Through each step, the new cells are created to the CYK table. First, when the system receives the stroke “2”, a cell (0, 0) is put into the CYK table as shown in Figure 5(1). Secondly, when the stroke “|” and then the stroke “-” from a symbol “+” are input, 2 new cells and 3 new cells are created as shown in Figure 5(2) and 5(3). When the stroke “8” is input, 4 new cells are created as shown Figure 5(4). Finally, after the user writes the stroke “6”, 5 cells are added to the CKY table as shown in Figure 5(4). Namely, the CYK table is updated from left to right when a new stroke marked red is input.

3.6. Multi-thread

Follow the flow of recognition shown in Figure 4, a user is required to wait until the current recognition process finishes in order to write a new stroke. It make the user feel uncomfortable and unnatural. As a result, to reduce the time delay in writing, the incremental recognition is performed on 2 threads.

All functions related to the interface are in thread 1, while thread 2 includes all functions in the recognition process. The user can write ME smoothly without waiting although our system has not finished the current recognition process for the latest input stroke yet.

4. Experiments

The following experiments are implemented on an Intel® Core i7-3770 CPU of 3.40GHz with 8GB memory. The OS is Window 7 Professional.

To evaluate our proposed incremental method, we use the dataset of 10,864 MEs that is collected from 62 elementary school children, 27 junior high school students and 26 members of our laboratory. We use 8,266 MEs for training and use 2,598 MEs for testing.

The first experiment is to measure the average waiting time for recognizing an ME by the batch method and the incremental method. Figure 6 shows the result. For the batch recognition, the waiting time is also the processing time. In the incremental method, however, the waiting time for an ME is the sum of the waiting time for all strokes.

After a new stroke i is written, the method performs the processing cycle from the segmentation step to the parsing step. The time to perform these steps is the processing time for stroke i, we call it r_i. In addition, there is elapsed time between two consecutive strokes, we call it after writing stroke i as e_i. From 2,520 MEs in another data set, the average elapsed time between two strokes is 0.7862 second. The waiting time for an ME is calculate as follows:

$$T_{incremental} = \sum_{i=1}^{n-1} f(r_i - e_i) + r_n$$

where $f(x) = x$ if $x>0$ otherwise 0. It is apparent from Figure 6 that the waiting time of the batch method rises up quickly when the number of strokes increases. On the contrary, the average waiting time of the incremental method increases slowly. Moreover, the average waiting time at stroke i (with i from 1 to 29) is less than average delay time (0.7862 second), so that the waiting time for an ME is also the elapsed time at the last stroke of that ME.

The second experiment is to measure the recognition rate. Table 1 shows recognition rates by the batch method, the incremental method and the tandem method, in which we start recognizing first j strokes by the batch method and switch to the incremental method to recognize later strokes. The incremental method and tandem method are implemented with multi-thread technique.

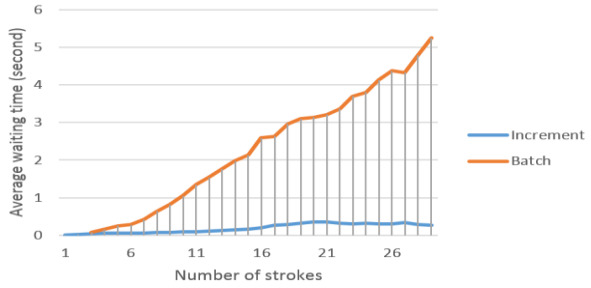

Figure 6: Average waiting time.

	Batch	Incre-mental	Tandem			
			j=3	*j=4*	*j=5*	*j=6*
Rate	63.78	62.74	63.55	63.36	63.47	63.66

Table 1: Recognition rates (%).

The incremental method keeps the recognition rate in comparison with the batch method without significant decrease. The tandem method improves the recognition rate from the incremental method and succeeds the small waiting time since the batch recognition employed for starting strokes takes small waiting time until j is less than 7 as shown in Figure 6. The difference of recognition rate between the batch method and the tandem method with j=6 is only 0.12 point.

Acknowledgement

This research is being supported by Grant-in-Aid for Scientific Research under the contract number (B) 24300095 and (S) 25220401. The authors would like to thank Professor Pham The Bao from Ho Chi Minh University of Science for helping us collect online mathematical expression dataset.

5. Conclusion

We have developed a method to recognize handwritten MEs in real-time. By updating the geometric features, recognizing new candidates and updating the CYK table after receiving every new stroke, the proposed method not only reduces the waiting time but also keeps the rate as high as the batch method. This has been confirmed by the experiments made on our dataset of 2,598 handwritten MEs. The experiments have suggested us to mix both the methods and the resultant mix method makes the waiting time as little as the incremental method and achieves recognition rate between the two methods.

References

[1] A. D. Le, T. V. Phan, and M. Nakagawa. A System for Recognizing Online Handwritten and Improvement of Structure Analysis. Proc. DAS, 51–55, 2014.

[2] M. Nakagawa, K. Machii, N. Kato, and T. Souya. Lazy Recognition as a Principle of Pen Interfaces. ACM INTERCHI, 89–90, 1993.

[3] S. Maclean, and G. Labahn. A new approach for recognizing handwritten mathematics using relational grammars and fuzzy sets. International Journal on Document Analysis and Recognition, 16(2):139–163, 2012.

[4] Knuth and Donald E. . The Art of Computer Programming Volume 2: Seminumerical Algorithms (3rd ed.). Addison-Wesley Professional, 501, 1997.

[5] K. Toyozumi, N. Yamada, T. Kitasaka, K. Mori, Y. Suenaga, K. Mase, and T. Takahashi. A study of symbol segmentation method for handwritten mathematical formula recognition using mathematical structure information. Proc. ICPR, 630–633, 2004.

[6] 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. Proc. ICDAR, 594–599, 2011.