Improving online handwritten mathematical expressions recognition with contextual modeling

Ahmad-Montaser Awal, Harold Mouchère, Christian Viard-Gaudin
IRCCyN/IVC – UMR CNRS 6597
Ecole polytechnique de l'université de Nantes - Nantes – France
{ahmad-montaser.awal, harold.mouchere, Christian.Viard-Gaudin }@univ-nantes.fr

Abstract

We propose in this paper a new contextual modelling method for combining syntactic and structural information for the recognition of online handwritten mathematical expressions. Those models are used to find the most likely combination of segmentation/recognition hypotheses proposed by a 2D segmentor. Models are based on structural information concerning the layouts of symbols. They are learned from a mathematical expressions dataset to prevent the use of heuristic rules which are fuzzy by nature. The system is tested with a large base of synthetic expressions and also with a set of real complex expressions.

1. Introduction

For scientists, nothing is better than modeling a given problem using mathematical notation. Almost all fields of science including human sciences use mathematics in less or more complex ways. To understand even more the importance of mathematical expressions (MEs), we have extracted all MEs from web pages of the French Wikipedia. Almost 77 000 expressions were found in 7000 web pages. Thus, MEs are universal communication tools among scientists. Furthermore, the tendency in scientific communities to use digital proceedings increased remarkably in the last few years. There are many tools to input MEs into digital documents. However, those tools require special skills to be used efficiently. Latex and MathML, for example, require knowledge of predefined sets of key words. Other tools, such as Math Type, depend on a visual environment to add symbols using the mouse and though needs lot of time.

Recent advances in the domain of digital pens and touch screens allow to widespread the use of handwriting input tools. Of course, these tools present an interesting alternative to input mathematical expressions into digital documents. Hence, it is essential to develop systems able to convert expressions from the natural handwritten way to a digital format. However, handwritten MEs recognition is more challenging than text recognition [1]. Unlike handwritten text which is a simple left to right sequence of characters; a ME is a complex 2D layout of mathematical symbols. The number of these symbol (~220 symbols) is by itself another challenge that requires powerful classifying tools. Moreover, the two dimensional layout causes many ambiguities in symbol roles, spatial relations, more examples can be found in [2][3]. Many researches have been done recently in this domain with promising results. Most of these researches consider expression recognition as a sequence of independent subtasks. This decomposition simplifies the problem, but errors inherited from one step cannot be easily corrected.

Our research focuses on the recognition of online handwritten MEs. The advantage of our proposition is to perform a simultaneous segmentation, recognition and interpretation of MEs under the restriction of a language model. Specifically, the classifier used to recognize symbols is based on a global learning method allowing to learn symbols directly from MEs performing at the same time the segmentation, and the interpretation.. The contribution of this paper is to propose a new method to model contextual information between symbols. Contextual models are learned directly from a ME database. These models serve not only in recognizing expression structure, but also in boosting the capacity of the symbols classifier by considering the n best class candidates.

2. State of the art

Generally, ME recognition takes place in three main steps [4]: segmentation, recognition and interpretation. Considering an online handwritten signal, the primitive unit, which allows to segment it, is a stroke. A stroke being a trace drawn between a pen down and a pen lift. However, in most cases a single symbol is composed of several strokes. Conversely, we will assume that one pen lift exists between consecutive symbols.

A good segmentation is the key point of a good recognition and interpretation. Hence the segmentation step consist in grouping strokes belonging to the same symbol. Early systems considered symbol segmentation as an independent step [5][6]. More recently, symbol segmentation and recognition are considered as one step. Thus, the segmentation is lead by symbol recognition [7][8][9], where recognition scores serve to choose groupings that are more likely to represent symbols. In order to decrease the complexity of this simultaneous optimization « best first search » [7], or CYK [9] algorithms are used.

The geometrical structure of a ME is usually more complex than that of a normal text. While a text is systematically written from left to right, math symbols can be written in almost all directions, see Figure 1. Therefore, MEs interpretation consists of analyzing geometrical structures of the expression and applying syntactic analysis. The objective of this interpretation is to find the derivation tree of the expression.

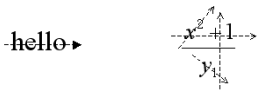


Figure 1 Writing directions in a normal text and a mathematical expression

Spatial relations between symbols are crucial for good interpretation. Even if all symbols are correctly segmented and recognized, a 2D analysis is required to correctly interpret the expression.

A method based on a "Definite Clause Grammar" is proposed in [4]. The efficiency of this DCG is increased by using left factored rules. More recently, Garain [10] proposed a context free grammar. A structure is built by dividing the expression recursively into horizontal and vertical bands. When reaching the level of atomic elements, grammar production rules are applied according to the type of spatial relations. In [11], the authors present an approach called "Fuzzy Shift-Reduce Parsing". This method uses a descending analysis assuring efficient verification. A probabilistic grammar has been proposed in [9]. Each production rule of the grammar is associated to a logical relation in addition to the probability of this rule. Thus, the recognition of an expression is transformed into a search of rules that maximize the probability of obtaining the result expression.

3. Proposed recognition system

The proposed architecture aims at handling the recognition of MEs as a simultaneous optimization of segmentation, symbol recognition and interpretation problems. The training of the system and also the recognition of expressions are done using the global architecture we proposed in [17], see Figure 2.

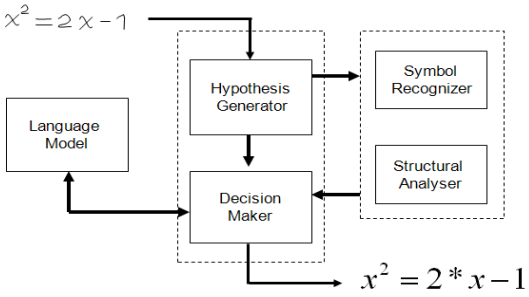


Figure 2 Expression recognizer architecture

The System is trained in two distinct stages in a global learning schema. Furthermore, the system is trained directly from mathematical expressions instead of training it with isolated symbols or using heuristic values for structural analysis. It takes into account the ground truth of the given ME (ideal segmentation and corresponding labels of symbols and spatial relations among those symbols) and the best current interpretation resulting from a specific segmentation, and corresponding recognized symbols. The architecture is detailed in the following sections.

3.1. Symbol hypothesis generator

The number of all possible segmentations is defined by the bell number. On a simple example shown in Figure 3, considering 7 strokes $B_7 = 877$ different segmentation. Bell numbers are calculated using the following recursive formula :

$$B_{n+1} = \sum_{k=0}^{n} \binom{n}{k} B_k \quad ; \quad \binom{n}{k} \text{ is a binomial coefficient.}$$

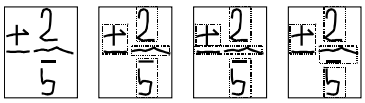


Figure 3 Example of grouping hypotheses

Hypothesis generator lists a number of possible combinations of strokes. Each group of strokes is called a symbol hypothesis (*sh*). From a computing perspective, it can be considered as a Dynamic Programming (DP) algorithm, which is well adapted to this kind of decision making problems [12]. However, the key point is that this is not a standard 1D-DP but we adopt an extension to a 2D-DP. To avoid the combinatory explosion of the search space, some constraints are added limiting the maximum number of strokes in one symbol and of hypothesis.

3.2. Symbol recognizer

The symbol recognizer provides, in addition to the best n candidates of each hypothesis, a recognition score of each candidate that will be used to calculate the recognition cost. We used a time delayed neural network (TDNN) [13] for its interesting properties of being insensitive to position shifts. However, an additional layer is added in order to convert the classifier outputs into probabilities. For a given hypothesis $p(c_j|sh_i)$ denotes the probability of the hypothesis sh_i being the class c_j ; where

$$\sum_j p(c_j|sh_i) = 1 \qquad (1)$$

Some methods considers N best candidate of the symbol classifier [9]. Similarly, others delay the decision of labeling ambiguous symbols to be resolved by the global context [7].

In order to determine the number of candidates retained for each sh_i. We retain k candidates with a max number N (k <=N). k is chosen under the condition: $\sum_{j=0}^{k-1} p(c_j|sh_i) \le Thresh$ to keep only candidates with high confidence and avoid to keep all N candidates systematically.

In the first training stage, the system is trained with the training expressions without considering contextual information. The goal of this stage is to train the symbol classifier. Furthermore, this stage gives the classifier the ability to identify wrong hypotheses during the segmentation using an additional class that we call the "junk class" [17].

3.3. Language model

A mathematical expression is produced by a 2D language. Context free grammars are efficiently used for parsing 1D languages (such as programming languages). However, parsing 2D languages is intractable and requires special algorithms and constraints to reduce complexity and ambiguity [3]. Graph grammar was introduced in [15], transferring parsing an ME to a graph rewriting problem.

Since two-dimensional grammars are faced to performance issues, we have described one as a set of one-dimensional rules on both vertical and horizontal axes. Vertical rules (VR) and horizontal rules (HR) are applied successively until elementary symbols are reached to perform a bottom up parsing algorithm.

Table 1 Example of a simple grammar

Rule	Relation type
sym ← x, y, 1, 2, …	
op ← +, -, x, …	
formule ← subExp op sym	(operator) [HR]
subExp ← subExp op sym	(operator) [HR]
subExp ← sym sym	(superscript) [VR]
subExp ← sym	

As shown in Table 1, each production rule of the grammar is associated to a spatial relation that describes the layout between the elements of the rule. On the other hand, each relation is associated to a cost function (see sections 3.4 and 3.5) that penalizes more or less the rule according to structural information of its elements. Ideal positions and sizes are difficult to be predefined because of the fuzzy nature of relations. Therefore, we try to model these relations instead of predefining them using some heuristic rules.

3.4. Structural analyzer

Structural analysis requires extracting spatial information of each symbol hypothesis to perform contextual evaluation. In order to avoid ambiguities, spatial information are extracted differently depending on the type of the recognized symbol as shown in Figure 4. Extracted information is the base line *y* and hypothesis height *h*. Furthermore, the spatial information of a produced sub-expression is calculated from its elements considering the spatial relation type.

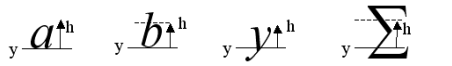


Figure 4 Structural information of different symbol types

When a relation R is built, the value of *y* and *h* of the corresponding sub-expression is computed from the N children components:

$$y_{SE} = f_R^y(y_{SE_1},...,y_{SE_N});\ h_{SE} = f_R^h(h_{SE_1},...,h_{SE_N})$$

The number of children N differs according to the type of the relation R; with N = {2, 3, 4}. For example, relations as “superScript” and “subScript" have two children. The “operator” relation has three children, while the “integral” relation has four.

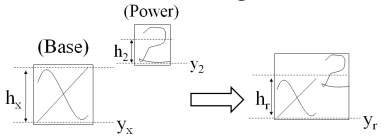
A sub-expression *(SE)* can denote three possible interpretations: a symbol hypothesis, a sub-expression, or the final result expression. Then, we define the normalized position and size differences *(dh, dy)* of a sub-expression *i* (SE_i) as follows:

$$dh_i = (h_{SE} - h_{SE_i}) / h_{SE} \; ; \quad dy_i = (y_{SE} - y_{SE_i}) / h_{SE}$$

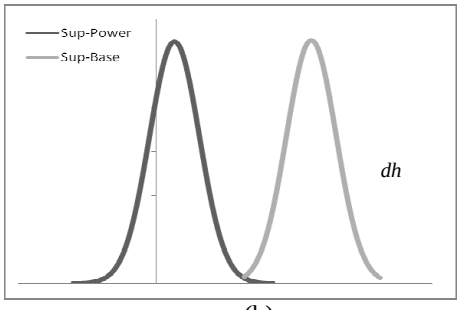
The second stage of the training aims at modeling spatial relations without considering the classifier. All occurrences of (dy, dh) of each sub-expressions for each kind of relations are computed using the ground truth of the training dataset of MEs. Then, two Gaussian models are constructed for each element of a relation. The first model reflects the pertinence of the size of this element for the sub-expression produced by the production rule associated to this relation, using dh_i. The second one reflects the quality of the alignment of the considered elements according to the produced sub-expression, using dy_i. Thus, for each relation R, 2N Gaussian models are constructed using the *n* occurrences of dh, dy of each sub-expression. The mean μ and variance σ of each model are calculated as follows (i = 1..*n*):

$$\mu_{SE,dx} = \frac{\sum_i dx_i}{n}; \quad \sigma_{SE,dx} = \sqrt{\frac{1}{n-1}\sum_i (dx_i - \mu_{SE,dx})^2}$$

For example, the relation “superscript”, which involves two elements, is associated to four models. Two models for size differences (*dh*) of each element, and another two for the alignment differences (*dy*).



(a)



(b)

Figure 5 (a) Illustration of the relation “superscript”, (b) Gaussian models of position differences (*dh*)

Figure 5 illustrates the structural information of the superscript relation and shows an example of Gaussian models of the position differences (*dh*).

Though, for a given situation we define the probability of a sub-expression being correctly positioned and sized regarding the produced SE by a Gaussian function:

$$g_{SE,dx}(x) = e^{-\frac{(x-\mu_{SE,dx})^2}{2\sigma^2_{SE,dx}}} \qquad (2)$$

Hence, the probability of a sub-expression being produced by the relation R is:

$$p(SE|R) = g_{SE,dh}(dh).g_{SE,dy}(dy) \qquad (3)$$

Back to the example of N sub-expressions producing a new sub-expression SE by the relation *R*. The probability that SE is produced by the N sub-expressions is:

$$p(SE|R) = \prod_{i=1}^{N} p(SE_i|R) \qquad (4)$$

But, from the definition of Bayes rule, the probability of a relation that produces a sub-expression knowing its Gaussian model is:

$$p(R|SE) = \frac{p(SE|R).p(R)}{p(SE)} \qquad (5)$$

but the term *p(SE)* can be ignored, because it is a constant for all relations, and by using the equation (4) we obtain the probability of a relation R:

$$p(R|SE) \propto \prod_{i=1}^{N} p(SE_i|R).p(R) \qquad (6)$$

where p(R) is the prior probability of the relation R calculated during the learning stage.

3.5. Decision maker

As shown in the example in Figure 6, a candidate expression produced by the grammar is represented by a relational tree. Each node represents a hypothesis of stroke grouping. The root of this tree contains in addition to the original ink signal: a proposed solution with its global recognition cost and the nature of the link with its children. In a similar way, each non terminal node represents a sub-expression, which contains the same information. Finally, terminals are symbol hypotheses proposed by the hypothesis generator with their recognition costs computed by the symbol recognizer.

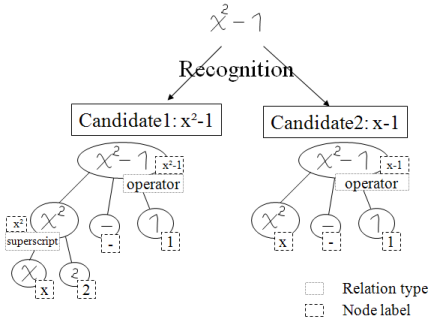


Figure 6 Relational trees of two candidates of a mathematical expression

Since recognition and structural scores are in form of probability (p), we use a logarithmic function to transform those scores into costs: $cost = -\log(p)$

The decision maker organizes all resulting relational trees, and selects the one that minimizes a global cost function. The cost of non-terminal nodes in the relational tree is defined by the recursive formula:

$$Cost(SE) = \begin{cases} Cost_{reco}(sh_i); if \text{ SE is a terminal} \\ \alpha.Cost_{struct}(R_{SE} | SE) + \sum_i Cost(SE_i); \text{Otherwise} \end{cases} \quad (7)$$

where: $Cost_{struct}(R_{SE} | SE) = -log(p(R_{SE} | SE))$ is the probability that the sub-expression SE is produced by the relation R. The cost of terminals is the recognition cost of the hypothesis sh_i:

$$Cost_{reco}(sh_i) = -\log(p(c_j | sh_i))$$

Finally, the global cost of a candidate expression with *n* symbols hypotheses linked by *r* relations is:

$$Cost(exp) = Cost(SE_{root})$$

Where "α" is a weighting factor between recognition and structural costs. The choice of alpha value in the equation (7) is very important to balance the differences between structural and recognition costs, it has been set experimentally to $\alpha = 0.18$.

4. Experiments

The expressions corpus is extracted from the base "Aster" [14]. A set of 36 expressions is chosen covering a majority of mathematical domains. Each expression contains in average 11 symbols, where the total number of distinct symbols is 34 classes. Each expression is artificially generated from a base of isolated symbols collected from 280 writers [16], therefore this dataset contains 10,080 mathematical expressions. In addition, each of those 36 expressions is written twice resulting a total of 72 expressions written by 10 writers, examples shown in Figure 7. Further details about databases can be found in [17].

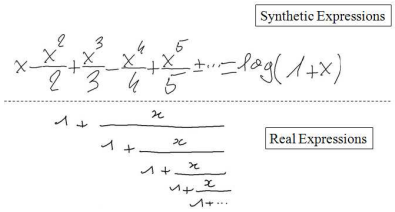


Figure 7 Examples of real and synthetic expressions

It is inappropriate to evaluate the system only at a global level, especially when dealing with long and complex expressions. In consequence, we have chosen three measures similar to those used recently in [7] and [9] in order to be more precise and to allow comparing our results. Denoting total number of symbols: N_{total}, we used the following measures:

SegRate=(correctly segmented symbols)/N_{total}
RecoRate=(correctly recognized symbols)/N_{total}
ExpRate= (correctly recognized expressions)/Exp_{total}

4.1. Results

The number of candidates considered as a result of symbol hypotheses classifying is an important factor. The curve Figure 8 shows expression recognition rate evolution on the synthetic DB when varying the average of considered candidates per symbol hypothesis.

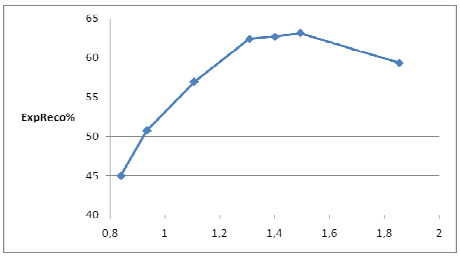


Figure 8 Evolution of test expressions recognition rate Vs average candidates number per symbol

This curve is obtained by fixing the max number of candidates to 5, and varying the threshold (see section 3.2). Considering only one recognition result for a hypothesis limits the classifier capacity. On the other hand, retaining a large number of candidates gives more freedom to the structure analyzer. This freedom allows to increase segmentation rate with a risk to overlap correct expression interpretation.

Thus, results in Table 2 are obtained by considering at maximum the first 5 candidates with the sum of their

recognition probabilities being below 0.95. We compare obtained results using this configuration and the contextual modeling proposed in this paper with those of the system proposed in [17] based on geometric modeling of spatial relations.

Table 2 Test Expression recognition rates

Synthetic DB	SegRate	RecoRate	ExpRate
Geometric modeling [17]	92.8	90.3	64.2
Gaussian modeling	91.4	88.7	64.9
Real DB	SegRate	RecoRate	ExpRate
Geometric modeling [17]	87.8	80.2	28.6
Gaussian modeling	83.9	76.2	27.1

Table 2 shows the interest of using Gaussian models for the contextual modeling. This latter increases expressions recognition rate on the synthetic DB from 64.2% to 64.9%. This improvement is not generalized on real expressions. ExpRate decreases from (28.6% to 27.9%) compared to 29.2% in the system proposed by Ha et al. [6], noticing that they use a different database and it is not possible to do a direct comparison of results. This general fall in performance compared to the geometric model comes principally from the strong dependence of learned models with the train database, which is synthetic. Thus, we assume that it is indispensable to use large database of real expressions in order to increase the generalization capacity of using Gaussian models.

5. Conclusion and perspective

We presented in this paper a new method for contextual modeling in a system for online handwritten mathematical expressions recognition. Contextual models are learned directly from ME database. Learned models participate in penalizing wrong locations of hypotheses. Another improvement comes from considering the best n candidates of the symbols classifier. We believe that the performance can be improved by integrating models of similar relations and ignore some others that cause ambiguities at relation level. The system has been tested with a large set of synthetic expressions and also on a set of real complex expressions with promising results.

References

[1] D. Blostein, A. Grbavec. Recognition of mathematical notation. *Handbook on Optical Character Recognition and Document Image Analysis*, Queen's university, World Scientific Publishing Company: 557-582, 1997.

[2] W. Martin. Computer input/output of mathematical expressions. 2nd Symp. on Symbolic and Algebraic Manipulations, New York: 78-87 1971.

[3] E.G. Miller, P. A. Viola. Ambiguity and Constraint in Mathematical Expression Recognition. 15th National Conference on Artificial Intelligence: 784-791 1998.

[4] K-F. Chan, D-Y. Yeung. An efficient syntactic approach to structural analysis of on-line handwritten mathematical expressions. Pattern Recognition 33: 375-384 2000.

[5] C. Faure, Z.X. Wang. Automatic perception of the structure of handwritten mathematical expressions. *Computer Processing of Handwriting*, World scientific, Singapore: 337-361, 1990.

[6] J. Ha, R.M. Haralick, I.T. Phillips. Understanding mathematical expressions from document images. 3rd ICDAR: 956-959, 1995.

[7] T-H Rhee, J-H Kim. Efficient search strategy in structural analysis for handwritten mathematical expression recognition. *Pattern Recognition 42(*12): 3192-3201, 2009.

[8] S. Smithies, K. Novins, J. Arvo. A Handwriting-Based Equation Editor. *the Graphics Interface*: 84-91, 1999.

[9] R. Yamamoto, S. Sako, T. Nishimoto, S. Sagayama. On-Line Recognition of Handwritten Mathematical Expressions Based on Stroke-Based Stochastic Context-Free Grammar. 10th IWFHR, La Baule, France: 249-254, 2006.

[10] U. Garain, B. Chaudhuri. Recognition of Online Handwritten Mathematical Expressions. IEEE Transactions on Systems, Man and Cybernetics 34: 2366-2376, 2004.

[11] J.A. Fitzgerald, F. G., T. K. Structural Analysis of Handwritten Mathematical Expressions Through Fuzzy parsing. 4th IASTED: 151-156, 2006.

[12] M. Held, R.M.K., The construction of discrete dynamic programming algorithms, *IBM Syst. J.* 4(2): 136-147, 1965.

[13] M. Schenkel, I. Guyon, D. Henderson. Online cursive script recognition using time delay neural networks and hidden Markov models. *Mach. Vis. Appl*., Special Issue on Cursive Script Recognition 8: 215-223, 1995.

[14] T.V. Raman. Audio system for technical readings. Cornell University, 1994.

[15] A Kosmala, G Rigoll, S Lavirotte, L Pottier. On-Line Handwritten Formula Recognition Using Hidden Markov Models and Context Dependent Graph Grammars. 5th ICDAR: 107-110, 1999.

[16] A.M. Awal, R. Cousseau, C. Viard-Gaudin. Convertisseur d'équations LATEX2Ink. 10th CIFED, Rouen, France: 193-194, 2008.

[17] A.M. Awal, H. Mouchère, C. Viard-Gaudin. Towards handwritten mathematical expression recognition. 10th ICDAR, Barcelona, Spain:1046-1050, 2009.