Recognition of On-line Handwritten Mathematical Expressions Using 2D Stochastic Context-Free Grammars and Hidden Markov Models

Francisco Álvaro, Joan-Andreu Sánchez, José-Miguel Benedí

Departamento de Sistemas Informáticos y Computación
Universitat Politècnica de València. Valencia, Spain
{falvaro,jandreu,jbenedi}@dsic.upv.es

Abstract

This paper describes a formal model for the recognition of on-line handwritten mathematical expressions using 2D stochastic context-free grammars and hidden Markov models. Hidden Markov models are used to recognize mathematical symbols, and a stochastic context-free grammar is used to model the relation between these symbols. This formal model makes possible to use classic algorithms for parsing and stochastic estimation. In this way, first, the model is able to capture many of variability phenomena that appear in on-line handwritten mathematical expressions during the training process. And second, the parsing process can make decisions taking into account only stochastic information, and avoiding heuristic decisions. The proposed model participated in a contest of mathematical expression recognition and it obtained the best results at different levels.

Keywords: Mathematical expression recognition; Handwriting recognition; Stochastic parsing; Symbol recognition; Structural analysis; Spatial relations; Hidden Markov models

1. Introduction

An essential part of information in science documents and many other fields is mathematical notation. Introducing mathematical expressions into computers and handling them usually requires special notation like LaTeX or MathML. However, in recent times there has been a great increasing of pen-based interfaces and tactile devices that allow users to provide handwritten data as input. This is a more natural way of introducing mathematical notation and it requires, in turn, developing systems that are able to recognize them. Recognition systems for mathematical expressions depend on the application [6]: on-line recognition systems for handwritten mathematical expressions, and off-line recognition systems for handwritten or printed mathematical expressions. In on-line recognition, the system input is usually a set of strokes that have geometric and temporal information. The system can take profit of temporal information of mathematical expressions that is not present in off-line recognition. This paper will be focused in on-line recognition of handwritten mathematical expressions.

Handwritten mathematical expression recognition can be divided into two major steps [9]: symbol recognition and structural analysis. Symbol recognition involves segmentation of the input strokes into mathematical characters and symbol classification of these hypotheses. Structural analysis deals with finding out the structure of the expression according to the symbols arrangement. Several methods have been proposed to solve these problems jointly and separately. If the symbol recognition and structural analysis problems are performed jointly, then the segmentation can be considered as a hidden variable. If the symbol recognition and structural analysis problems are carried out separately, then the segmentation is done explicitly. The second approach is usually computational less expensive but it is prone to segmentation errors. The first approach is computational more expensive and efficient A* search strategies must be defined [29].

Several proposals have been studied for printed symbol recognition based on Support Vector Machines [24], Nearest Neighbor techniques or Hidden Markov Models (HMM) [3]. A survey of methods is provided in [9] for on-line recognition. The most usual representation techniques for on-line recognition are based on template or structural matching and HMM. Currently the error rate of mathematical symbols is about 5% for the best known classification techniques [14, 22, 31]. A very low error rate in mathematical symbol segmentation is crucial because it largely contributes to a reduction in the recognition of the full expression.

Structural analysis of mathematical expression has been carried out by considering syntactic models that are able to represent spatial relations between the mathematical symbols. Trees are the mostly accepted models and they have been considered in [8, 12, 28, 40, 42] to represent these 2D spatial relations in on-line recognition. Minimum spanning trees have also been considered [13]. These tree structures are obtained through a parsing process. Evaluation and comparison of the structural analysis in mathematical expression recognition has hitherto been difficult because: first, most of the proposals used private data, and second, there has been a lack of standard performance evaluation measures [5, 19]. Lately, some public large corpora have been developed [23, 32] and also new metrics have been proposed [1, 30, 41]. Recently a Competition on Recognition of Online Handwritten Mathematical Expressions[1] (CROHME 2011) has been proposed [25]. This competition tried to define an experimental setup that allowed the comparison of different systems.

The contribution of this paper is two-fold. First, from theoretical point of view, this paper takes a step forward in mathematical expression recognition by introducing a stochastic formal model based on 2D stochastic context free grammars (SCFG). The main contribution of this paper is an adequately formalized model that allowed us to deal appropriately with the two fundamental problems associated to this kind of models, that is, the learning problem, and the interpretation problem. The learning problem deals with the learning of the structural and stochastic part of the model from data. This paper introduces the learning of the main stochastic distributions of the model. The interpretation problem deals with obtaining the structure of the mathematical expression. The algorithm for parsing mathematical expressions according to the proposed model is also defined in this paper. It

[1]Competition held in *International Conference on Document Analysis and Recognition* (ICDAR 2011). `http://www.isical.ac.in/~crohme/`

is important to remark that a correct formalization of the model allows us to use classical probabilistic estimation algorithms for learning and efficient algorithms for stochastic parsing.

Second, from a more applied point of view, this paper contributes to mathematical symbol recognition and spatial relations classification as follows. HMM have been widely used for symbol recognition, in this work we applied two well-known sets of features used in handwritten text recognition [34, 35] that have not been yet tested in mathematical symbol recognition. Regarding spatial relations classification, we extended the work presented in [4] for printed mathematical expressions. We propose adding new features to account for variability in the handwritten case, as well as to recognize spatial relations that were not considered in that paper. We also employed a different method for computing the geometric features.

A system for handwritten mathematical expression recognition based on parsing 2D-SCFG is completely detailed in this paper. It is composed by hybrid HMM as mathematical symbol classifier and spatial relations are determined according to geometric features. This model is able to solve jointly the problems involved in mathematical expression recognition: symbol recognition and structural analysis. The described system participated in the CROHME 2011 competition previously mentioned and it obtained the best results.

The remainder of the paper is organized as follows. First, a review of related works is given in Section 2. Then, an overview of the on-line handwritten mathematical expressions recognition system is presented in Section 3. The formal statistical framework based on a two-dimensional extension of SCFG is described in Section 4. The mathematical symbol recognition process is explained is Section 5 and the spatial relations classification is detailed in Section 6. Finally, Section 7 presents a set of experiments performed to validate this approach using a public database of mathematical expressions and also the results obtained in the CROHME 2011 competition are reported. Conclusions and future work are presented in Section 8.

2. Related work

There are several papers with detailed surveys on on-line recognition of mathematical expressions [9, 14, 37]. In this section we will briefly describe those papers that have influenced at most the approach of this paper.

There are two main steps in mathematical expression recognition: symbol segmentation and recognition and structural analysis of the recognized symbols. On-line handwritten mathematical expressions are usually represented as a set of strokes. The segmentation problem consists on grouping properly these strokes to produce mathematical symbols. The segmentation problem has been tackled by computing connected components in [28] or applying the projection profile cutting method in [27], and the obtained strokes are then classified. The primary unit representation in mathematical expression recognition can be mathematical symbols [39] or strokes. When using strokes as primary unit representation, the segmentation into strokes can be explicitly used [42, 14, 40, 37, 29] or not [31, 22]. The use of explicit segmentation into strokes introduces the problem of grouping them to compose symbols, and it makes the search difficult. Given that currently there are powerful symbol segmentation techniques [22], in this paper, the primary unit representation was mathematical symbols.

Recognition of handwritten mathematical symbols is carried out using classical techniques. Several methods have been proposed to solve this problem, such as HMM [39, 18, 16], Neural Networks [33], Elastic Matching [7, 36] or Support Vector Machines [17]. Furthermore, some of these proposals combine on-line and off-line information to perform hybrid classification and improving recognition results [39, 17]. An important advantage of using HMM is that they may not require an explicit segmentation for training [31, 22]. In this work we have used HMM and we have combined on-line and off-line characteristics with HMM.

Several approaches have tackled the structural analysis problem by using formal grammars. Chou [12] proposed to use SCFG in order to recognize printed mathematical expressions. Other proposals have been presented using definite clause grammars [10] or graph grammars [21]. Yamamoto *et al.* [40] presented a statistical formulation for on-line parsing of handwritten mathematical expressions based on the Cocke-Younger-Kasami (CYK) algorithm. Our proposal is similar to the approach based on two-dimensional SCFGs that was described in Průša and V. Hlaváč [28]. Průša and V. Hlaváč used weights associated to regions that contained symbols and then these regions were combined according to a penalty function based on region weights and syntactic rule weights. Our proposal uses stochastic distributions associated to regions in a similar way to [40], but the stochastic distributions associated to the model were different in our proposal. The stochastic definition of our model allowed us to use classical parsing and estimation algorithms for SCFG [20].

3. System Overview

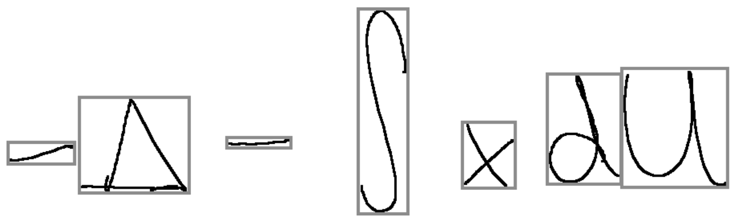
This section presents an overview of the developed system, and the following sections provide a detailed description of the most important parts of the model.

As we mentioned in the previous section, the symbol segmentation process was separated from the recognition of the full expression. That is, the segmentation process was considered as an explicit variable and not as a hidden variable. Considering the symbol segmentation as a hidden variable would be also possible in the developed system. The image was rendered by using linear interpolation between the points belonging to the same stroke. Then the connected components of the image were computed to obtain and a set of segmentation hypotheses (see Fig. 1).

The recognition of the full expression was carried out with a Two-Dimensional Stochastic Context-free Grammar (2D-SCFG)[2]. This 2D-SCFG is introduced as a modification of a SCFG for strings. But in the two-dimensional case, the SCFG must account for spatial relations. Several spatial relations are stochastically explored between two regions: horizontal, subscript, superscript, vertical and inside. This information is coded in the rules.

[2]Matrices and left scripts (${}^{a}_{b}x$) were not considered in this paper.

Strokes input	Rendered image

10
27
9028 6150
9039 6155
9044 6152
...
26
9579 5788
9576 5774
9582 5772
...
41
9582 5744
9582 5741
...
	

	10 strokes input sequence 7 connected components ($\Delta = 3$ strokes and $x = 2$ strokes)

Figure 1: Mathematical expression initial segmentation by computing the connected components in the rendered image of the input sequence of strokes.

Then, a stochastic constraint that takes into account the geometric position of each region is introduced in the parsing process.

Note that two important problems are, first, the obtaining of the 2D-SCFG, and second, the parsing process. In this research, the rules of the grammar and the rule probabilities were manually defined. It is important to point out that the stochastic estimation of the rule probabilities could be performed with classic stochastic estimation algorithms [20] by defining appropriate algorithms. Section 4 introduces a parsing algorithm from which *inside* and *outside* versions could be derived. The other probability distributions of the 2D-SCFG were stochastically estimated from examples.

In the parsing process, the initialization step was done using a mathematical symbol classifier to recognize the set of segmentation hypotheses. Afterwards, the productions of the grammar defined a set of syntactic and spatial constraints that guided the stochastic parsing process in order to build a complete structure of the most probable expression.

The mathematical symbol recognition step was performed by using both on-line and off-line information. Note that the segmentation strategy based on computing connected components is not enough to solve this problem. Touching characters detection was not tackled in this work. However, the system is able to properly recognize symbols that are composed by more than one connected component. The detection of these type of symbols was integrated in the parsing process. For instance, an equal symbol (=) was recognized because the grammar had a production that composed an equal symbol as the vertical concatenation of two horizontal lines. In addition, some extra classes had to be added to recognize symbols whose components were not actual terminal symbols of the grammar, as the body of the i or a single stroke representing the '*co*' part of the cosine function. Then, an i symbol is recognized as the vertical concatenation of a dot $(\cdot)$ and the body of an i $(\imath)$, as well as a cosine function can be recognized as the horizontal concatenation of '*co*' and the letter '*s*'.

Finally, using the global information of the whole expression structure helped to improve the initial decisions made in segmentation, symbol recognition and spatial relations among regions.

4. Two-dimensional Stochastic Context-free Parsing

SCFGs are a powerful formalism of syntactic pattern recognition that have been extensively used for string patterns. However, it is possible to slightly modify this formalism in order to model two-dimensional problems. In this paper, we studied the modeling of mathematical expressions using SCFGs. Hence, a 2D extension of a SCFG and its corresponding version of the CYK parsing algorithm are defined below.

4.1. Two-dimensional Stochastic Context-free Grammars

First, we present the notation for Two-Dimensional Stochastic Context-free Grammars (2D-SCFGs) that constitute the formal framework used in this work.

A *Context-Free Grammar* (CFG) G is a four-tuple (N, Σ, S, P), where N is a finite set of non-terminal symbols, Σ is a finite set of terminal symbols ($N \cap \Sigma = \emptyset$), $S \in N$ is the starting symbol of the grammar, and P is a finite set of rules: $A \to \alpha$, $A \in N$, $\alpha \in (N \cup \Sigma)^+$. A CFG in Chomsky Normal Form (CNF) is a CFG in which the rules are of the form $A \to BC$ or $A \to a$ (where $A, B, C \in N$ and $a \in \Sigma$).

A *Stochastic Context-Free Grammar* (SCFG) $\mathcal{G}$ is defined as a pair (G, p), where G is a CFG and $p : P \to]0, 1]$ is a probability function of rule application such that $\forall A \in N : \sum_{i=1}^{n_A} p(A \to \alpha_i) = 1;$ where n_A is the number of rules associated to non-terminal symbol A.

A 2D-SCFG for mathematical expression recognition is a generalization of a SCFG, and introduces some significant differences. First, the terminal symbols are the recognized mathematical symbols in the two-dimensional regions corresponding to the segmentation hypotheses obtained in the pre-processing stage. Second, the grammar rules have an additional parameter that describes the spatial relation among regions. This relation is defined as $A \xrightarrow{spr} \alpha$, where *spr* denotes the spatial relation that models the rule. Common spatial relations for mathematical expression recognition are: *horizontal*, *vertical*, *inside*, *subscript* and *superscript*. Terminal productions do not contain the spatial relation parameter. Finally, the main difference lies in how to combine the grammar rules to account for an image. We now describe a stochastic parsing method for this type of two-dimensional models.

4.2. CYK parsing for 2D-SCFG

Let $\mathcal{G}$ be a CNF 2D-SCFG, and let $\mathcal{C} = \{\ c_i^{<x,y>} \mid i : 1 \ldots C\ \}$ be the set of segmentation hypotheses (representing the input expression) provided after the pre-processing step, where $c_i^{<x,y>}$ represents the segmentation hypothesis located at region defined by points $<x, y>$ (top-left and bottom-right corners, respectively).

In order to compute the probability of the best parsing of the input expression (represented by $\mathcal{C}$) by the grammar $\mathcal{G}$ is necessary to define a new version of the well-known CYK-based Viterbi algorithm [26] to consider two-dimensional images.

Following [15], this algorithm is essentially a dynamic programming method, which is based on the construction of a parsing table $\mathcal{L}$. Each element in $\mathcal{L}$ is defined according to the following concepts. Value $e^{<x,y>}[A]$ is the probability that A is solution of the mathematical expression contained in the region defined by $<x,y>$. Analogously, $e_l^{<x,y>}[A]$ is the probability that A is solution of the mathematical subexpression contained in the region $<x,y>$, considering l segmentation hypotheses. Finally, $\mathcal{L}_l = \{\ e_l^{<x,y>}[A]\ \}$ is a stochastic parse structure where each element $e_l^{<x,y>}[A]$ is composed from l segmentation hypotheses.

Following [26, 15], we now present a stochastic parsing algorithm for 2D-SCFG based on the well-know CYK algorithm. This algorithm will allow us to calculate the parsing table $\mathcal{L}$. As in [26, 15], we will divide this process into two steps. First, the initialization begins building the set $\mathcal{L}_1$ from the set of segmentation hypotheses defined in $\mathcal{C}$:

$$\mathcal{L}_1 \ = \ \mathcal{L}_1 \ \cup \ \{\ e_1^{<x_i,y_i>}[A]\ \} \qquad\qquad \forall i : 1 \dots C$$

such that:

$$e_1^{<x_i,y_i>}[A] = \max_m \ \{p(A \to m)\ q_m(c^{<x_i,y_i>})\}$$

where m is a particular mathematical symbol, for which there is a specific recognizer. Value $q_m(c^{<x_i,y_i>})$ is the probability provided by the HMM symbol classifier (see section 5) of this class m for segmentation hypothesis $c^{<x_i,y_i>}$.

Next, the parsing process continues calculating new subproblems of increasing size. Formally, the general case is computed as

$$\mathcal{L}_l \ = \ \mathcal{L}_l \ \cup \ \{\ e_l^{<x,y>}[A]\ \} \qquad\qquad \forall\ l : 2 \dots C$$

with:

$$e_l^{<x,y>}[A] = \max_{B,C}\ \max_{r}\ \max_{k:1\dots l-1}\ \max_{\substack{e_k^{<x_B,y_B>}[B] \in \mathcal{L}_k \\ e_{l-k}^{<x_C,y_C>}[C] \in \mathcal{L}_{l-k}}}$$

$$(p(A \xrightarrow{r} BC)\ e_k^{<x_B,y_B>}[B]\ e_{l-k}^{<x_C,y_C>}[C]\ p_r(<x_B,y_B>,<x_C,y_C>))$$

where a new subproblem $e_l^{<x,y>}[A]$ is created from two subproblems of minor size $e_k^{<x_B,y_B>}[B]$ and $e_{l-k}^{<x_C,y_C>}[C]$ taking into account both the syntactic constraints, defined by $p(A \xrightarrow{r} B\ C)$, and the following constraints: first, r represents the spatial relation, and second, $p_r(<x_B,y_B>,<x_C,y_C>)$ is the probability that both regions were arranged according to the spatial relation r (see Section 6).

Looking at the 2D CYK algorithm, the first remarkable difference is that the parsing table is indexed by only one index. On the standard CYK parsing algorithm two indexes explain the positions that define some substring. In the 2D case, there is a level for each subproblem size, and these levels store a set of elements which contain their two-dimensional space information. For that reason, at the initialization step the built subproblems are added at $\mathcal{L}_1$, that is to say that they cover one input element (in this case a connected component).

After that, the parsing process continues by building new subproblems of increasing size, where both spatial and syntactic constraints are taken into account for each new subproblem.

Finally, the time complexity of the algorithm is $O(n^4|P|)$ where n is the symbol number of the input expression and $|P|$ the rule number of the 2D-SCFG. The time complexity of the classical CYK is $O(n^3|P|)$. However, this complexity was reduced to $O(|P|n^3 \log n)$ by using a spatial data structure as is described in [2].

5. Mathematical Symbol Recognition

Hidden Markov Models (HMM) were used in this paper to perform the mathematical symbol recognition process by using two well-known sets of features widely employed in handwritten text recognition [34, 35]. In on-line recognition of mathematical symbols, the basic input unit is usually a sequence of strokes. In our approach, the basic input unit were isolated symbols made up from strokes. The path described by the points in each stroke was rendered, and then mathematical symbol hypotheses were obtained. The mathematical symbols were got by classifying a segmentation hypothesis according to off-line characteristics. Then each segmentation hypothesis was reinforced by combining both on-line and off-line information. Note that this process could be done in the opposite direction: first a segmentation hypothesis could be obtained according to on-line characteristics, and then each hypothesis could reinforced by combining both on-line and off-line information.

The following subsections describe the recognition process for both cases and also the combination of both modalities.

5.1. Classification with off-line characteristics

A sequence of strokes describes a path into the space, hence it is possible to draw the image representation from the on-line input. In this approach, we used simple linear interpolation between every two consecutive points (see Fig. 2).

Each segmentation hypothesis was made up by computing the connected components in the rendered representation of the input sample. Then, each connected component was classified by using HMMs. HMM have been widely used for mathematical symbols classification in on-line recognition [14]. However their use in off-line recognition has not received much attention. In this paper, we explored the technique described in [34] applied to off-line mathematical symbols.

The rendered representation was adequately normalized to a fixed size row, keeping the aspect ratio (see Fig. 2.a). Then, the image was transformed into a sequence of fixed-dimension feature vectors as follows: the image was divided into a grid of small square cells, sized a small fraction of the image height. Each cell was characterized by the following features (Fig. 2): normalized grey level (b), horizontal grey-level derivative (c) and vertical grey-level derivative (d). To obtain smoothed values of these features, feature extraction was extended to a 5×5-cell window centered at the current cell and weighted by a two-dimensional Gaussian function in b) and a unidimensional Gaussian function in c) and d). The derivatives were computed by least squares fitting a linear function. Columns of cells were processed from left to right and a feature vector was built for each column by stacking

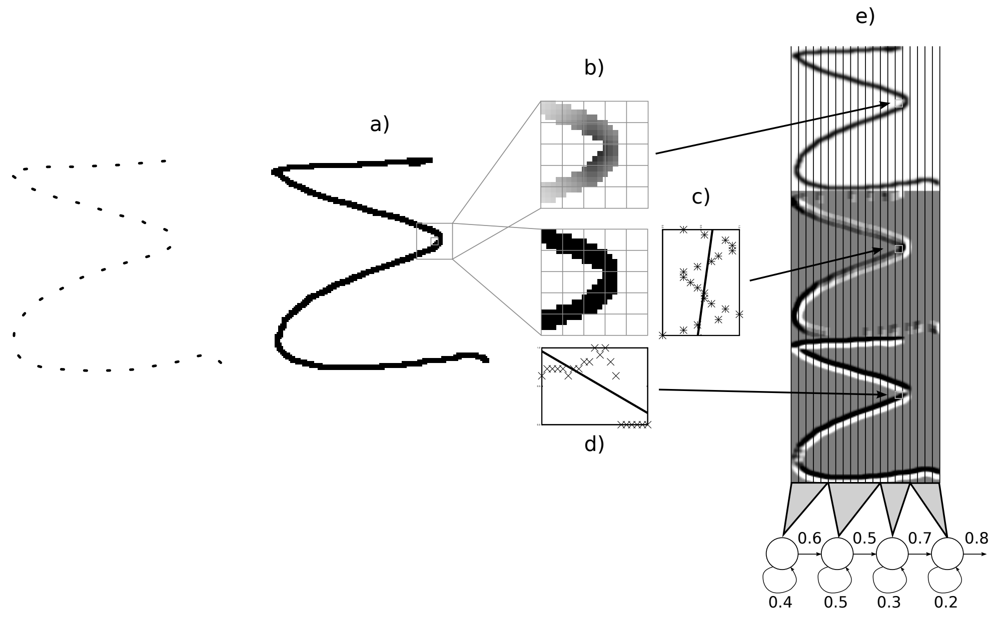


Figure 2: Rendering example of a certain sequence of strokes through linear interpolation and off-line feature extraction.

the features computed in its constituent cells. Fig. 2.e) shows a graphical representation of the obtained values.

5.2. Classification with on-line characteristics

Given a set of strokes that represents a segmentation hypothesis according to the off-line classification, HMMs were used to classify the sample into one of the possible mathematical symbol classes considering on-line information. The input set of points was processed to obtain a sequence of feature vectors as is described in [35]. First, a preprocessing step was applied to the input, and this process involves four steps: repeated points elimination, noise reduction, writing speed normalization and size normalization. Afterwards, some features were computed for each point (x_t, y_t). In this case, each vector was composed by 15 features: normalized horizontal and vertical position (x_t, y_t), normalized first derivatives (x'_t, y'_t), second derivatives (x''_t, y''_t), curvature k_t, and the four lowest frequency Fourier components of the *Discrete Fourier Transform* where each component had real and imaginary part. The number of states of each symbol HMM was estimated depending on the average number of feature vectors of the training samples.

5.3. Hybrid classification

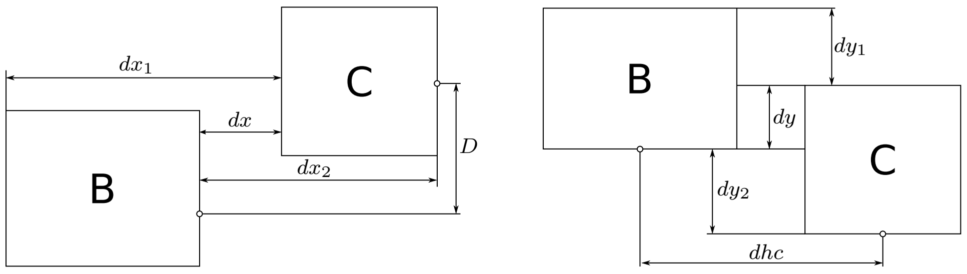
Finally, both on-line and off-line HMMs were combined in order to take advantage of all the information that was available. A naive Bayes classifier was used (assuming uniform class priors), which aimed at balancing the relative reliability of the on-line (x) and off-line (y) models by using a weight parameter (α):

$$\Pr(x, y|c) = \alpha \cdot \Pr(x|c) + (1 - \alpha) \cdot \Pr(y|c) \tag{1}$$

In order to obtain the posterior probability of a given sample from both on-line and off-line models, we had to compute the posterior probability in both cases. These probabilities can be efficiently computed with the well-known *forward-backward* algorithm [38]. It should be noted that this calculation requires a scaling factor parameter, so the hybrid classifier had 3 parameters to be tuned: α for the combination of probabilities, β_{on} scaling factor for on-line classifier and β_{off} scaling factor for off-line classifier.

6. Spatial Relations Classification

Given two 2D regions B and C, the relation between them was statistically determined. In this paper, five different spatial relations were considered: *horizontal* (BC), *subscript* (B_C), *superscript* (B^C), *vertical* ($\frac{B}{C}$) and *inside* ($\sqrt{C}$). The relation between two regions was represented as a single numerical vector according to several geometric features. We used two features H and D previously defined for classification of spatial relations in printed mathematical expressions [4], and 7 additional features (see Fig. 3). These new features aimed at improving classification results by providing more information given that hand-written expressions present more variability than printed expressions. Feature H represents the relative size between symbols, and feature D the difference of their vertical centers. Values dx, dx_1 and dx_2 extended the horizontal position information, as well as dy, dy_1 and dy_2 extended the vertical position information. The difference between the horizontal centers (dhc) helped to recognize the vertical relation, where this distance should be small. Moreover, using these new features allowed us to also classify the *inside* relationship, whereas features H and D were insufficient. Finally, every feature value was normalized by the height of the parent region in the relation (h_B), which made these features invariant to the scale of the mathematical expression coordinates.



$$h_B = \text{height}(B); \quad H = \frac{\text{height}(C)}{h_B}$$

$$D = \frac{\text{center}_{\text{ver}}(B) - \text{center}_{\text{ver}}(C)}{h_B}; \quad dhc = \frac{\text{center}_{\text{hor}}(B) - \text{center}_{\text{hor}}(C)}{h_B}$$

$$\text{features} = [H, D, dhc, \frac{dx}{h_B}, \frac{dx_1}{h_B}, \frac{dx_2}{h_B}, \frac{dy}{h_B}, \frac{dy_1}{h_B}, \frac{dy_2}{h_B}]$$

Figure 3: Spatial relations geometric features for automatic classification.

Given a set of training mathematical expressions we extracted a set of samples of several spatial relation types represented by the described features. Then, we trained a Support

Vector Machine (SVM) classifier using these samples. In this way, we automatically learnt the spatial relations distribution from training data, and no heuristic decision was made. From the resulting trained SVM models, we were able to compute the posterior probabilities as is described in [11].

A very important issue for the performance of the spatial relations symbol classifier was the way that the vertical center ($\text{center}_{\text{ver}}$) was calculated. Horizontal center ($\text{center}_{\text{hor}}$) was computed as $(left + right)/2$. From the defined relationships, the main problem is to distinguish between *horizontal*, *subscript* and *superscript* cases. These three relations can be very ambiguous even for human beings and for that reason it is crucial to compute a proper vertical center for each region.

Aly *et al.* [4] computed the vertical center as the center after normalizing the bounding box by adding a virtual ascender or a virtual descender or both, depending on the character category. A different method was used in this paper. First, four types of symbols were defined: *ascendant* (d, t, λ), *descendant* (y, p, μ), *normal* $(x, -, \theta)$ and *middle* $(7, L, \Pi)$. In the symbol recognition step the vertical centroid of the symbol was modified according to this classification. If the class was not *normal*, then the centroid was modified. For class *ascendant* the centroid was displaced down to $(centroid + bottom)/2$. Likewise, if type was *descendant* the centroid was displaced up to $(centroid + top)/2$. Finally, if the type was *middle*, then the geometric vertical center was considered $(top + bottom)/2$. Fig. 4 shows an example of the different centers computed for a mathematical symbol according to its type.

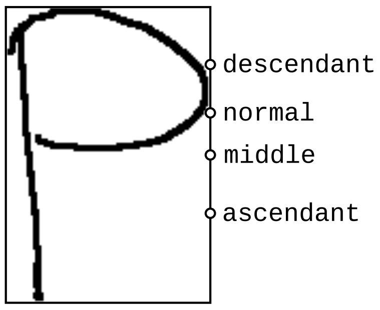


Figure 4: Example of vertical center computation for a mathematical symbol depending on its type.

Once the vertical centers were calculated for every symbol, this information was hierarchical inherited as follows. The combination of two regions B and C resulting in a new region A had to follow some rules in order to preserve good center values. A decision was made depending on the spatial relation between them and the rule of the grammar, giving rise to different cases. In general, for the *subscript* relation (B_C) and the *superscript* relation (B^C) the center of their combined region A is the center of region B (Fig. 5a), for the *inside* relation ($\sqrt{C}$) the resulting center is the center of C (Fig. 5b), and for the *horizontal* relation ($B\,C$) the final center is computed as $(\text{center}_{\text{ver}}(B) + \text{center}_{\text{ver}}(C))/2$ (Fig. 5c). The *vertical* relation ($\frac{B}{C}$) depended on the rules of the grammar. For example, for parsing a fraction the following two rules were used:

$$\text{Fraction} \xrightarrow{vertical} \text{Expression} \quad \text{OverExp}$$
$$\text{OverExp} \xrightarrow{vertical} \text{HorzLine} \quad \text{Expression}$$

where in the second rule $\text{center}_{\text{ver}}(\text{OverExp}) = \text{center}_{\text{ver}}(\text{HorzLine})$ and in the first rule $\text{center}_{\text{ver}}(\text{Fraction}) = \text{center}_{\text{ver}}(\text{OverExp})$. As a result, the center of the fraction is the horizontal line center.

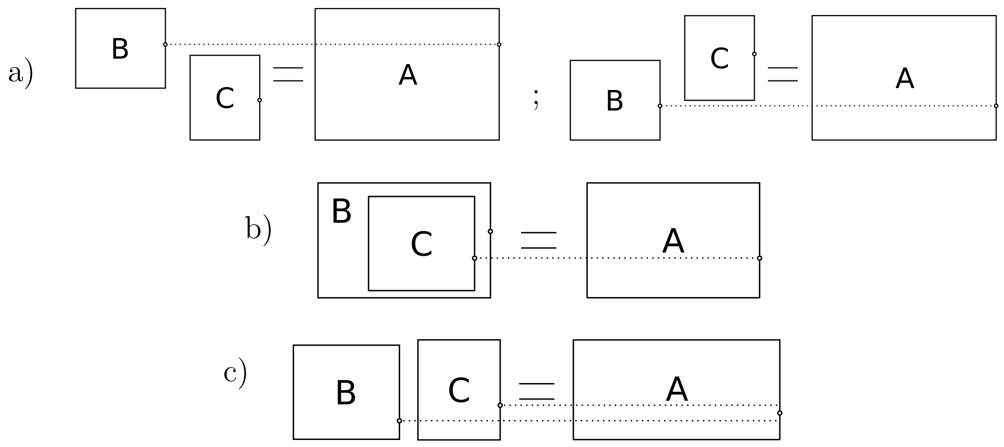


Figure 5: Vertical center computation of the combination of two regions according to the spatial relation among them: a) subscript/superscript; b) inside; c) horizontal.

Finally, in order to achieve a better normalization of spatial features, the height of the region resulting after the combination of two regions is computed according to the applied rule. This is because the geometric features are normalized by the height of the parent region. Analogously to vertical center computation, the height of a combined region is the height of the region whose center is inherited (base in subscript/superscript relation, content in square root). For horizontal relation, the resulting region has the rightmost region's height, and vertical relation does not inherits this information.

7. Experiments

A full model for on-line recognition of handwritten mathematical expression has been described. In this section, we describe the experiments that were carried out in order to validate the developed system. First, each part of the system is tested individually: mathematical symbol recognition and spatial relations classification. Then, the whole system is evaluated by recognizing mathematical expressions. Experimentation was carried out by using MathBrush[3] [23], a large public database of on-line handwritten mathematical expressions. This database was recently made publicly available and there are not comparable

[3]http://www.scg.uwaterloo.ca/mathbrush/corpus

results with other systems. Therefore, results of the CROHME 2011 competition are also reported. The described system participated in this competition, hence, comparable result are also provided.

Preliminary experiments as well as mathematical expression recognition experimentation were carried out by using the MathBrush corpus [23]. This database contains $4,654$ annotated mathematical expressions written by 20 different writers. The number of mathematical symbols is 26K and they are distributed in 100 different classes. We split the database into 5 groups of writers with 4 writers on each group in order to perform a cross-validation scheme and analyzing the performance according to writer dependencies. Writer dependent and writer independent experiments were carried out, and for each scenario 5 recognition experiments were performed as a result of the cross-validation strategy, and then the average results were obtained.

7.1. Mathematical Symbol Recognition

As Section 5 explained, we used a hybrid mathematical symbol classifier based on HMM. The on-line and off-line models had some parameters to be tuned, as well as the combination of both classifiers. For both the writer dependent scenario and writer independent scenario, the parameters were tuned only for one of the cross-validation partitions. For each experiment, the annotated symbols were extracted from the set of mathematical expressions. Once these parameters were properly adjusted, the behaviour of the linear combination of the HMM classifiers was studied (see Section 5.3). Fig. 6 represents the relative improvement with respect to the best recognition result (on-line or off-line) for one of the performed experiments, according to a weight parameter α (see Exp. (1)).

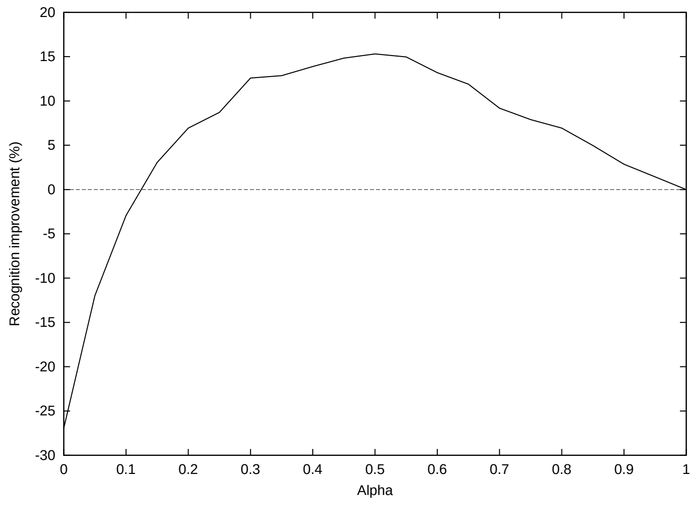


Figure 6: Relative improvement of hybrid mathematical symbol classification according to α weight (writer dependent).

The final results of the mathematical symbol recognition experiments are shown in Table 1. Results showed that on-line recognition obtained higher recognition rate than off-line recognition, and hybrid recognition significantly improved the classification accuracy. As it was expected, symbol recognition rate notably decreased for writer independent experiments.

Table 1: Mathematical symbol recognition: recognition rate and relative improvement.

Writer	Rec_{ON}	Rec_{OFF}	Rec_{HYB}	Improvement
Dependent	85.63	81.84	88.14	17.47
Independent	78.64	76.98	83.81	24.20

MathBrush database has been developed recently, hence, there are not many published results and for that reason comparison is difficult. Mathematical symbol recognition experiments were carried out in [16] also using the MathBrush corpus. However, that work was focused only on symbol recognition. It did not perform a writer dependent or writer independent experiments. Different training and test sets were defined and the number of classes was reduced to 93. The final reported recognition rate was 82.9% by using an on-line HMM mathematical symbol classifier.

Keshari *et al.* [17] presented an hybrid SVM mathematical symbol classifier using on-line and off-line features. They used a different nonpublic database and the number of classes was 137. In that work, they obtained a relative improvement of 10% for writer dependent experiments, and a 5% relative improvement for writer independent experiments. It should be noted that we achieved greater improvement in independent case, in contrast to the results presented in [17]. We thought that it is due to some writers had significant differences in their writing style, and for that reason the on-line classifier did not obtain very good results in writer independent experiments. Hence, the hybrid classifier took more advantage of the combination because both classifiers were more balanced.

7.2. Spatial Relations Classification

For each cross-validation partition, a SVM with a Gaussian kernel was trained and then the test set was classified. Table 2 shows the results obtained for each scenario, where the recognition rate is reported for each type of spatial relation and using two different sets of features (see Section 6). According to the results, the incorporation of new geometric features has demonstrated to improve the recognition rate as well as to properly classify the *inside* relationship.

In this case, spatial relations had less dependencies to writer style than symbol shapes given that the classification results were very similar in all cases. The global accuracy was around 96% in average, but superscript and subscript relations presented a high error. It should be noticed that there was a significant difference in the amount of samples of each spatial relation. In the test set horizontal relations were very common (70%), whereas inside relations were infrequent (4%). Vertical relations were about 11% and subscript and superscript relations were about 6% and 9% respectively. Furthermore, the grammar syntactic information will help to improve spatial relations recognition.

Table 2: Recognition rate in spatial relation classification.

#Feat	Writer	Hor.	Subscript	Super	Vertical	Inside	Total
2	Dep.	96.29	44.39	72.55	92.25	6.31	86.94
	Ind.	96.43	46.34	70.17	92.61	3.57	85.97
9	Dep.	98.56	74.43	91.30	97.40	99.33	96.32
	Ind.	98.54	73.90	89.17	96.55	99.46	95.81

7.3. Mathematical Expression Recognition: MathBrush database

Finally, mathematical expression recognition experiments were performed. The HMM classifier for mathematical symbol recognition and the SVM for spatial relations classification were trained as previously discussed. The 2D-SCFG was manually defined to cover general mathematical expressions, taking into account the type of expressions of the corpus.

Evaluation of mathematical expression recognition systems is a difficult problem [5, 19], and several metrics have been proposed [30, 41, 1]. The EMERS metric [30] is a dissimilitude value computed as the tree edit distance between the tree representation of the obtained expression and the reference expression, ant it is not normalized. The main problem of the EMERS metric is that the representation ambiguity of coded mathematical expressions can produce different trees of the same expression. In [41], the authors proposed a representation that seems to cope with this ambiguity and they also presented their corresponding performance metrics. However, the representation required precise information of the structure of the expression at stroke level, and the MathBrush corpus annotation had not completely and explicitly this information. Recently, we proposed IMEGE[4] metric, a global measure based on image representation, which also deals with ambiguity. This metric is based on the image representation of the two expressions [1]. It computes a global dissimilitude value normalized in the range $[0, 100]$ such that if two expressions are identical the corresponding IMEGE value is equal to zero.

Table 3 shows the results of the mathematical expression recognition experiment, where several measures were used: symbol segmentation rate (SYM_{seg}), symbol recognition rate for well segmented symbols (SYM_{rec}), expression recognition rate (EXP_{rec}), and global expression metrics EMERS and IMEGE. Expression recognition rate was computed as the percentage of expressions whose IMEGE value was equal to zero. As expression recognition rate is a very pessimistic measure, we also report the rate of expressions for which the structure is perfectly recognized without regarding the symbol recognition errors (Struct). Results show that the proposed system is able to properly recognize handwritten mathematical expressions. The obtained performance evaluation measures validates the model, where all its parts are learnt automatically from training samples. Despite that the segmentation strategy of this approach is simple and it could be improved [22], symbol segmentation rate was 85.12% for writer dependent experiment and 83.84% for writer independent experiments.

[4]Software available at `http://users.dsic.upv.es/~falvaro/imege.html`

Table 3: Mathematical expression recognition results with MathBrush database.

Writer	SYM_{seg}	SYM_{rec}	EXP_{rec}	Struct	EMERS	IMEGE
Dep.	85.12%	86.49%	19.39%	44.10%	4.98	37.51
Ind.	83.84%	84.11%	19.15%	43.22%	5.14	40.08

The symbol recognition approach based on hybrid HMM also presented a good behaviour obtaining symbol recognition rate of 86.49% and 84.11% for writer dependent and writer independent scenarios, respectively. Expression recognition rate was about 19% and taking into account only the structure of the expressions the recognition rate increases up to 44%. It should be noted that writer dependent results are better that writer independent results, as it was the expected behaviour. Finally, global comparison is difficult because, as far as we know, there are not results presented over this database using these measures. For that reason, in the following section we present comparable results of the developed system in a recent competition.

7.4. Mathematical Expression Recognition: CROHME competition

The described system participated in the recent Competition on Recognition of Online Handwritten Mathematical Expressions [25]. Given that the comparison of mathematical expression recognition systems is not an easy task [5, 19], this competition was a good opportunity to compare different approaches in identical conditions. Five research groups participated, although system 5 was developed by the organizers and it did not compete.

A training set and and two parts were defined for the competition. Part-I contains 296 expressions and Part-II consists of 921 expressions. Part-I was a subset of Part-II. The number of symbol classes in Part-I was 37 whereas 57 terminal symbols were considered in Part-II. Likewise, the expressions in Part-II of the training set were more complex than the Part-I expressions. Finally, a hidden test set was used to evaluate the performance of each system participating in the competition. The test set for Part-I contained 181 expressions and the test set for Part-II was composed of 348 expressions including the expressions in Part-I.

The reported metrics are: stroke classification rate (ST_{rec}), symbol segmentation rate (SYM_{seg}), symbol recognition rate for well segmented symbols (SYM_{rec}), and expression recognition rate (EXP_{rec}). First three metrics regarding symbol recognition are well-defined, and EXP_{rec} was computed considering that an expression was perfectly recognized if the labeled tree and the recognized tree were identical. This measure can be biased because the same expression can be represented by different trees.

Every model of the system was trained using the defined training sets for each part: hybrid HMM for mathematical symbol recognition and SVM for spatial relations classification. Moreover, the competition also provided a grammar for each part, which we slightly adapted to a proper 2D-SCFG definition for our model. Table 4 shows the final results of the competition [25], where we participated as system number 3.

Table 4: CROHME 2011 evaluation results [25].

	System	ST_{rec}	SYM_{seg}	SYM_{rec}	EXP_{rec}
Part-I	1	53.23	59.06	88.78	4.42
	2	22.39	27.98	82.11	0.55
	3	**78.73**	**88.07**	**92.22**	**29.28**
	4	37.41	55.15	81.71	0.00
	5	78.57	87.56	91.67	40.88
Part-II	1	51.58	56.50	91.29	2.59
	2	22.11	28.25	83.76	0.29
	3	**78.38**	**87.82**	**92.56**	**19.83**
	4	52.28	78.77	78.67	0.00
	5	70.79	84.23	87.16	22.41

According to the results of the competition, the developed system had a good performance and it is robust as the results barely varied from Part-I to Part-II. It obtained significantly better results than the other participants for symbol recognition metrics. Regarding expression recognition rate, although the organizers' system result was higher, our system also obtained good results and was declared winner of the competition.

8. Conclusions

In this paper we presented an on-line handwritten mathematical expression recognition system. We defined a formal statistical framework based on 2D-SCFG and its corresponding CYK-based parsing algorithm. We also described a complete system where all components could be learnt from training data. Finally we performed several experiments over a large database and we concluded the following:

- The combination of on-line and off-line information significantly improves mathematical symbol recognition.
- Spatial relations between regions can be efficiently modeled using geometric features. Thus, a statistical distribution can be learnt from training data and then heuristic decisions could be avoided.
- The statistical framework based on 2D-SCFG is able to properly recognize handwritten mathematical expressions. Furthermore, structural information can help to improve the initial symbol recognition and spatial relations decisions.

For future work, given the statistical framework based on 2D-SCFG, the spatial relations classification should be improved. More sophisticated segmentation techniques should be incorporated in order to improve segmentation accuracy and to be able to tackle the touching characters problem. Other features could be used to train HMM for mathematical symbol recognition.

Acknowledgements

Work supported by the EC (FEDER/ FSE) and the Spanish MEC/MICINN under the MIPRCV “Consolider Ingenio 2010” program (CSD2007-00018), the MITTRAL (TIN2009-14633-C03-01) project, the FPU grant (AP2009-4363), and by the Generalitat Valenciana under the grant Prometeo/2009/014.

References

[1] F. Álvaro, J.A. Sánchez, and J.M. Benedí. IMEGE: Image-based Mathematical Expression Global Error. *DSIC-PRHLT Technical Report, Universitat Politècnica de València*, 2011. http://hdl.handle.net/10251/13084.

[2] F. Álvaro, J.A. Sánchez, and J.M. Benedí. Recognition of printed mathematical expressions using two-dimensional stochastic context-free grammars. In *International Conference on Document Analysis and Recognition (ICDAR)*, 2011.

[3] F. Álvaro and J.A. Sánchez. Comparing several techniques for off-line recognition of printed mathematical symbols. In *Proceedings of ICPR*, volume 0, pages 1953–1956, Istambul, Turkey, August 2010.

[4] W. Aly, S. Uchida, A. Fujiyoshi, and M. Suzuki. Statistical classification of spatial relationships among mathematical symbols. In *Proceedings of ICDAR*, pages 1350–1354, july 2009.

[5] A.-M. Awal, H. Mouchère, and C. Viard-Gaudin. The problem of handwritten mathematical expression recognition evaluation. In *International Conference on Frontiers in Handwriting Recognition (ICFHR), 2010*, pages 646–651, nov. 2010.

[6] D. Blostein and A. Grbavec. *Handbook of Character Recognition and Document Image Analysis*, chapter 21, Recognition of Mathematical Notation, pages 557–582. World Scientific Publishing Company, 1997.

[7] K.F. Chan and D.Y. Yeung. Elastic structural matching for online handwritten alphanumeric character recognition. In *Pattern Recognition, 1998. Proceedings. Fourteenth International Conference on*, volume 2, pages 1508 –1511 vol.2, aug 1998.

[8] K.F. Chan and D.Y. Yeung. An efficient syntatic approach to structural analysis for on-line handwritten mathematical expressions. *Pattern Recognition*, 33:375–384, 2000.

[9] K.F. Chan and D.Y. Yeung. Mathematical expression recognition: a survey. *International Journal on Document Analysis and Recognition*, 3:3–15, 2000.

[10] K.F. Chan and D.Y. Yeung. Error detection, error correction and performance evaluation in on-line mathematical expression recognition. *Pattern Recognition*, 34(8):1671 – 1684, 2001.

[11] C.C. Chang and C.J. Lin. LIBSVM: A library for support vector machines. *ACM Transactions on Intelligent Systems and Technology*, 2:1–27, 2011.

[12] P. A. Chou. Recognition of equations using a two-dimensional stochastic context-free grammar. In W. A. Pearlman, editor, *Visual Communications and Image Processing IV*, volume 1199 of *SPIE Proceedings Series*, pages 852–863, 1989.

[13] Y. Eto and M. Suzuki. Mathematical formula recognition using virtual link network. In *Proceedings of ICDAR*, pages 762–767, Washington, DC, USA, 2001. IEEE Computer Society.

[14] U. Garain and B.B. Chaudhuri. Recognition of online handwritten mathematical expressions. *IEEE Trans. on Systems, Man, and Cybernetics - Part B: Cybernetics*, 34(6):2366–2376, 2004.

[15] J. Goodman. Semiring parsing. *Computational Linguistics*, 25(4):573–605, 1999.

[16] L. Hu and R. Zanibbi. HMM-based recognition of online handwritten mathematical symbols using segmental K-means initialization and a modified pen-up/down feature. In *Proceedings of ICDAR*, September 2011.

[17] B. Keshari and S.M. Watt. Hybrid mathematical symbol recognition using support vector machines. In *Proceedings of ICDAR*, volume 2, pages 859 –863, sept. 2007.

[18] A. Kosmala, G. Rigoll, S. Lavirotte, and L. Pottier. On-line handwritten formula recognition using hidden markov models and context dependent graph grammars. In *Proceedings of ICDAR*, pages 107 –110, sep 1999.

[19] A. Lapointe and D. Blostein. Issues in performance evaluation: A case study of math recognition. In *Proceedings of ICDAR*, pages 1355–1359, Washington DC, USA, 2009. IEEE Computer Society.

[20] K. Lari and S.J. Young. The estimation of stochastic context-free grammars using the inside-outside algorithm. *Computer Speech and Language*, 4:35–56, 1990.

[21] S. Lavirotte and L. Pottier. Mathematical formula recognition using graph grammar. In *Proceedings of the SPIE*, volume 3305, pages 44–52, 1998.

[22] Z.X. Luo, Y. Shi, and F.K. Soong. Symbol graph based discriminative training and rescoring for improved math symbol recognition. In *Proceedings of ICASSP*, pages 1953–1956, 2008.

[23] S. MacLean, G. Labahn, E. Lank, M. Marzouk, and D. Tausky. Grammar-based techniques for creating ground-truthed sketch corpora. *Int. J. Doc. Anal. Recognit.*, 14:65–74, March 2011.

[24] C. Malon, S. Uchida, and M. Suzuki. Mathematical symbol recognition with support vector machines. *Pattern Recognition Letters*, 29(9):1326–1332, 2008.

[25] H. Mouchère, C. Viard-Gaudin, U. Garain, D.H. Kim, and J.H. Kim. CROHME2011: Competition on recognition of online handwritten mathematical expressions. In *Proceedings of ICDAR*, September 2011.

[26] H. Ney. Stochastic grammars and pattern recognition. In P. Laface and R. De Mori, editors, *Speech Recognition and Understanding. Recent Advances*, pages 319–344. Springer-Verlag, 1992.

[27] M. Okamoto and B. Miao. Recognition of mathematical expressions by using the layout structures of symbols. *Proceedings of ICDAR*, pages 242–250, 1991.

[28] D. Průša and V. Hlaváč. Mathematical formulae recognition using 2d grammars. *Proceedings of ICDAR*, 2:849–853, 2007.

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

[30] K. Sain, A. Dasgupta, and U. Garain. EMERS: a tree matching-based performance evaluation of mathematical expression recognition systems. *IJDAR*, pages 1–11, 2010.

[31] Y. Shi, H. Li, and F.K. Soong. A unified framework for symbol segmentation and recognition of handwritten mathematical expressions. In *Proceedings of ICDAR*, pages 854–858, 2007.

[32] M. Suzuki, F. Tamari, R. Fukuda, S. Uchida, and T. Kanahori. Infty- an integrated OCR system for mathematical documents. In C. Vanoirbeek, C. Roisin, and E.Munson, editors, *Proceedings of ACM Symposium on Document Engineering*, pages 95–104, Grenoble, 2003.

[33] A. Thammano and S. Rugkunchon. A neural network model for online handwritten mathematical symbol recognition. In D.S. Huang, K. Li, and G. Irwin, editors, *Intelligent Computing*, volume 4113 of *LNCS*, pages 292–298. Springer Berlin / Heidelberg, 2006.

[34] A.H. Toselli, A. Juan, and E. Vidal. Spontaneous handwriting recognition and classification. In *Proc. of the 17th International Conference on Pattern Recognition*, pages 433–436, Cambridge, UK, August 2004.

[35] A.H. Toselli, M. Pastor, and E. Vidal. On-line handwriting recognition system for tamil handwritten characters. In J. Martí, J.M. Benedí, A. Mendonça, and J. Serrat, editors, *Pattern Recognition and Image Analysis*, volume 4477 of *Lecture Notes in Computer Science*, pages 370–377. Springer Berlin / Heidelberg, 2007.

[36] B.Q. Vuong, Y. He, and S.C. Hui. Towards a web-based progressive handwriting recognition environment for mathematical problem solving. *Expert Syst. Appl.*, 37:886–893, January 2010.

[37] B.Q. Vuong, S.C. Hui, and Y. He. Progressive structural analysis for dynamic recognition of on-line handwritten mathematical expressions. *Pattern Recognition Letters*, 29:647–655, 2008.

[38] F. Wessel, R. Schluter, K. Macherey, and H. Ney. Confidence measures for large vocabulary continuous speech recognition. *IEEE Trans. Speech and Audio Processing*, 9(3), Mar 2001.

[39] H.J. Winkler. HMM-based handwritten symbol recognition using on-line and off-line features. In *Acoustics, Speech, and Signal Processing, 1996. ICASSP-96. Conference Proceedings., 1996 IEEE International Conference on*, volume 6, pages 3438–3441, may 1996.

[40] R. Yamamoto, S. Sako, T. Nishimoto, and S. Sagayama. On-line recognition of handwritten mathematical expressions based on stroke-based stochastic context-free grammar. *IEIC Technical Report*, 2006.

[41] R. Zanibbi, P. Amit, H. Mouchère, C. Viard-Gaudin, and D. Blostein. Stroke-Based Performance Metrics for Handwritten Mathematical Expressions. In *Proceedings of the 11th International Conference on Document Analysis and Recognition*, Beijing, China, September 2011.

[42] R. Zanibbi, D. Blostein, and J.R. Cordy. Recognizing mathematical expressions using tree transformation. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 24(11):1–13, 2002.