Pattern Recognition 48 (2015) 837–848

Contents lists available at ScienceDirect

Pattern Recognition

journal homepage: www.elsevier.com/locate/pr

Matching based ground-truth annotation for online handwritten mathematical expressions

Nina S.T. Hirata *, Frank D. Julca-Aguilar

Department of Computer Science, Institute of Mathematics and Statistics, University of São Paulo, 05508-090 Rua do Matão, 1010 São Paulo, Brazil

ARTICLE INFO

Article history:
Received 25 September 2013
Received in revised form
8 September 2014
Accepted 18 September 2014
Available online 28 September 2014

Keywords:
Structural pattern analysis
Shape information
Handwriting recognition
Linear assignment problem
Ground-truth annotation
Mathematical expression dataset

ABSTRACT

Assessment of mathematical expression recognition at expression level only is not sufficient to diagnose strengths and weaknesses of different recognition systems. In order to make assessment at different levels possible, large datasets annotated with ground-truth data at different levels, such as at symbol segmentation, symbol classification, symbol/sub-expression spatial relationships, baselines or whole expression levels, are needed. Creation of ground-truthed datasets of handwritten mathematical expressions is a challenging task due to the need to cope with a large variability of symbol classes, expression layouts, writing styles, among other issues including the fact that manual annotation is an error-prone procedure. We propose an expression matching approach where symbols in a transcribed expression are assigned to the corresponding symbols in the respective model expression. Matching is formulated as a simple linear assignment problem where matching cost is defined as a weighted linear combination of local (symbol) and global (structural) characteristics. Once a symbol-to-symbol assignment is computed, not only symbol labels but all other ground-truth data attached to the model expression can be automatically transferred to the transcribed expression. We use two independent large test sets to empirically evaluate the influence of the cost function terms on matching performance. Results show mean symbol assignment rates above 99% on both sets, suggesting the potential of the method as an useful tool for helping the creation of ground-truthed online mathematical expression datasets.

1. Introduction

With the advent of tablet like devices, there is an increasing interest in online recognition of handwritten mathematical expressions (MEs). Interesting and useful applications of online recognition of MEs include numerous possibilities, notably those related to inputting mathematical notation into computer systems. In the last years, active research has been carried out in all aspects related to online recognition of handwritten MEs, including symbol segmentation and recognition [1–4], structural analysis [5–7,2], integrated approaches for recognition [8–12], and recognition evaluation [13–17]. Overview of ME recognition approaches and issues can be found in [18–20].

In the field of pattern recognition, a difficulty related to evaluating recognition performance is the scarcity of large and public datasets annotated with ground-truth information. Authors frequently consider their own datasets, making difficult the tasks of reproducing reported results and performing comparison among different methods. Efforts to address this issue have recently started to appear in the ME recognition community [11,13,21,22] and, following tendencies in the field, Competition on Recognition of Online Handwritten Mathematical Expressions (CROHME) has been recently established [23–25]. From the recognition rates reported in the CROHME editions, one can conclude that full recognition of MEs is still a challenging task. In order to advance development in this field, larger and diverse datasets for evaluating different aspects of recognition would be welcome. A comprehensive list of recognition evaluation related issues is given in [26], and importance of correct definition of ground-truth data and metrics for system evaluation are discussed in [16].

Often, for training and evaluating symbol recognition algorithms, users are required to enter several samples of each symbol, a tiring and boring task. From the user point of view, writing full expressions is more natural than writing several symbols individually and repeatedly. Thus, at an early stage of this research, while dealing with symbol recognition algorithms, we decided to collect samples of full expressions and then extract symbol samples from the expressions, rather than collecting samples of individual symbols. Moreover, it is likely that compared to the

* Corresponding author.
E-mail addresses: nina@ime.usp.br (N.S.T. Hirata), faguilar@ime.usp.br (F.D. Julca-Aguilar).

http://dx.doi.org/10.1016/j.patcog.2014.09.015
0031-3203/

symbols that are written individually, those obtained in this way will better resemble the symbols that actually occur in expression samples. A side effect of this approach is, however, the need to segment and label individual symbols in the sample expressions.

With respect to the problem of labeling individual symbols in handwritten MEs, some preliminary results obtained with an expression matching-based method were reported in [27]. The method assumes that the first expression is a model, with symbols previously segmented and labeled manually, and the second one is a transcription of the first, with symbols correctly segmented but not labeled. Then, the matching process associates each symbol in the transcribed expression to the corresponding symbol in the model expression, allowing automatic labeling of symbols in the transcribed expression. Segmentation is performed during writing, i.e., whenever the time gap between two strokes is larger than a user controlled threshold, they are separated. A bounding box enclosing the set of strokes considered as being part of one symbol is drawn dynamically, inducing users to undo the last written stroke whenever a non-desired stroke is joined to a previous symbol (see more details in [28]).

Given that current symbol classifiers perform very well [24], one could argue that labeling of individual symbols could be performed using one of those classifiers. However, one advantage of the matching approach is the fact that it also allows transferring of structural level ground-truth from one expression to the other. In contrast, only individual symbol class identification would not be sufficient for structural correspondence because a same symbol may occur repeatedly within an expression.

One of the contributions of this work relates to this observation. We propose a general framework for the creation of ground-truthed online ME datasets. The matching method is one of the main steps of this framework. The formulation of expression matching problem as a linear assignment problem presented here extends the method described in [27], including additional local features for the matching cost computation. The proposed framework, with discussions on how some other methods proposed in the literature for the creation of online ME datasets are related to it, and the details of the matching formulation are presented in Section 2.

At this point it is noteworthy to mention that our method does not make any specific assumptions related to context. It only assumes that the "objects" to be matched (in our case, mathematical expressions) are bidimensional structures composed by atomic units (in our case, individual symbols) and that they are at a similar scale and also spatially aligned. For instance, the symbol features for the definition of matching cost are expressed by means of shape dissimilarity (and not considering a specific symbol classification method). Hence, although the target objects in this work are mathematical expressions, the proposed framework and matching method could be adapted for chemical equations or even some types of 2D diagrams.

In Section 3, we detail the metrics used to evaluate matching performance and present a thorough evaluation of the proposed method on two large independent datasets, our own dataset (ExpressMatch dataset [29]) and the MfrDB dataset [22]. Matching results show that an overall mean symbol assignment rate superior to 99% is achieved. One aspect of special interest in this work is the evaluation of the influence of structural and symbol cost terms in matching performance. In Section 4 we examine some poor performing matching pairs and list some common types of symbol assignment errors together with a discussion on why they occur and how some of them could be fixed. Finally, in Section 5 we present the conclusions and point some future work.

2. Proposed matching approach

Since correct ME recognition requires not only correct symbol recognition but also correct understanding of the spatial arrangement among symbols, to improve overall ME recognition rate it is necessary to assess and understand where recognition is failing and how different techniques perform at different recognition levels. To that end, it is important to thoroughly experiment methods and techniques on large and statistically representative datasets. In order to automate experimental evaluation, datasets annotated with ground-truth data at different levels of ME structure are required.

Based on the idea of matching expressions discussed in the Introduction, we propose the following general procedure for generating samples of ground-truth annotated MEs:

1. creation of a corpus of model sample expressions, with correctly segmented symbols and ground-truth data attached to them;
2. capturing of samples (input expressions) of the models by having users transcribing them;
3. segmentation of input expression symbols;
4. matching of input expression symbols to the corresponding ones in the model; and
5. transferring of ground-truth data from the model to the input expression.

2.1. Model expression creation and symbol segmentation

In step 1, model expressions can be generated using grammars that describe MEs as in [13], having the advantage that ground-truth of each model expression is known. However, defining grammars is not a simple task and some expressions may, even being syntactically correct, correspond to expressions that semantically are unnatural. Another way to create model expressions is by hand selecting them. Hand selection of models presents advantages related to an easier control of the creation process, allowing a corpus of model expressions to be built in such a way as to be statistically representative of a given domain, with types of expressions, symbols, notations and respective frequencies specified to follow the distribution observed in that domain. In this case, however, there is a need to manually create them and attach ground-truth data.

In step 2, images rendered from the LaTeX representation of model expressions (either directly generated by a grammar or hand generated) can be shown for transcription. Alternatively, handwritten expression images can be generated just by handwriting the expressions or by rendering them by composing handwritten individual symbols, as in [11]. However, mimicking an actual handwritten expression by composing individually written symbols (possibly by distinct individuals) is not simple. If model expressions are handwritten, then there is a need to manually segment the symbols.

Segmentation of symbols can be performed by a specific segmentation algorithm, or based on approaches that perform ink data capture and symbol segmentation simultaneously. In either case, an interactive segmentation correction procedure may be very helpful.

In this work, we hand-select expression models and handwrite them to be shown for transcription. Segmentation, both in model expressions and in transcriptions, is performed during ink capture as detailed in [28]. After symbols in a transcribed expression are correctly segmented, expression matching is applied to establish the symbol-to-symbol correspondence.

Alternatively, both segmentation and matching could be carried simultaneously as in [13]. Subsets of strokes are evaluated with respect to its likelihood of being the strokes of one symbol, and relationships between several neighboring subsets of strokes are analyzed in order to match a subset of strokes to a terminal symbol in the model expression generated by a grammar. The

approach in [13] also differs to ours with respect to matching; we consider matching of 2D input expression to a 2D model expression, while [13] considers matching of a 2D input expression to an 1D description of a 2D model expression.

2.2. Matching formulated as an assignment problem

In the proposed matching method, we assume that both model and transcribed expressions are handwritten and that symbols are individually segmented. In addition, we also assume that model and input expressions are spatially and scale aligned.

The problem of assigning each symbol in the transcribed (input) expression to the corresponding symbol in the respective model expression will be referred to as *expression matching*. This problem can be formulated as a simple linear assignment problem as follows. Suppose that for each pair of symbols (s_m, s_i), with s_m in the model expression and s_i in the input expression, a matching cost $c(s_m, s_i)$ is associated. The goal is to select a one-to-one assignment between symbols in both expressions, with minimum cost. The problem can be solved using, for instance, the well-known Hungarian algorithm [30]. Notice that the cubic order complexity of the Hungarian algorithm is not really an issue in our case because the average number of symbols in the expressions is relatively small.

2.2.1. Graph representation of expressions

We model expressions as graphs where symbols correspond to vertices and spatial relationship between symbols are expressed by the edges. Both vertices and edges may encode information such as coordinate, angle and size values, as well as symbol and structural features. We consider complete graphs, which imposes no preference regarding the neighborhood considered and thus may be able to capture both local and global structural information.

An ideal cost function is one that associates small costs to the pairs corresponding to correct assignments and high costs otherwise. A natural choice of a cost function should consider both symbol features and structural information, such as spatial relationships with respect to neighboring symbols. Matching cost formulation is presented in detail in Section 2.2.3.

2.2.2. Spatial and scale alignment

Different instances of a same expression written by different users or at different moments may differ in spatial location as well as in scale and rotation. Because of the nature of the matching cost function considered in this work, it is important that both expressions are spatially and scale aligned, before matching cost computation.

To that end, first the position of the vertices is computed as being the centroid of the respective symbol stroke coordinates. The center of the bounding box of all vertices is taken as the center of the expression graph. Input graph vertices are translated so that the graph center matches the center of the model graph. Then, the bounding box of the input graph is scaled independently in width and in height in such a way as to match those of the model graph. During the scaling process, the position of each vertex in the input graph is also scaled proportionally within the box. It is assumed that rotation is negligible. Fig. 1 shows a pair of aligned model-input expressions. Symbol size is not scaled since only scale-invariant symbol features are considered.

2.2.3. Matching cost

In order to define a matching cost between a vertex v_m in the model expression graph and a vertex v_i in the input expression graph, we use a measure inspired from [31] that describes the model graph deformation due to replacing vertex v_m with v_i. The rationale behind this is that since both graphs correspond to two instances of a same expression, it is reasonable to expect a good spatial registration between them when they are spatially and scale aligned. Hence, deformation in the model graph would be very small when one of its vertices v_m is replaced with the corresponding one in the input graph, whereas deformation would be large if v_m were replaced with any other arbitrary vertex of the input graph.

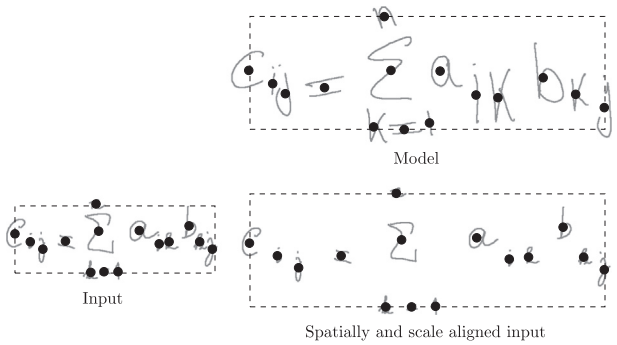

Fig. 1. Spatial and scale alignment of the input expression graph to the respective model expression graph (see details in the text). Vertices, represented by black circles, and the bounding box of the graph vertices are overlaid on the corresponding expressions for visualization.

In a general form, a matching cost function that captures the idea described above can be expressed as a composition of two cost terms: a cost term c_v related to vertex attributes and a cost term c_e related to edge attributes:

$$c = \alpha c_v + (1-\alpha) c_e \tag{1}$$

with $0 \le \alpha \le 1$ and $0 \le c_v, c_e \le 1$.

Edge cost: The edge cost should be able to capture structural deformations. These can be characterized by differences in the neighborhood relationship. Specifically, we consider deformation of the model graph when one of its vertices, v, is replaced with a vertex u in the input graph, as proposed in [31]. Let $E(v)$ denote the set of all edges incident to v. Let $e = (v, v') \in E(v)$ denote one of such edges and let $\tilde{e} = (u, v')$. Then difference between e and $\tilde{e}$ can be used to characterize the deformation, as in the cost [31] given by

$$c_e(v,u) = \frac{1}{\#E(v)} \sum_{e \in E(v)} c_s(e, \tilde{e}) \tag{2}$$

where

$$c_s(e,\tilde{e}) = \beta \frac{|\cos\theta - 1|}{2} + (1-\beta)\frac{||e| - |\tilde{e}||}{C} \tag{3}$$

and $\#E(v)$ is the cardinality of set $E(v)$, θ is the angle between e and $\tilde{e}$, $|e|$ is the length of e, and C is a constant equal to the length of the bounding box diagonal of the model expression graph, used to normalize the second term. Fig. 2 illustrates the edge cost concept.

Vertex cost: To express the dissimilarity between features encoded in two vertices, we propose the use of shape context [32], both at local (symbol) and global (expression) levels. In ME recognition context, shape context has recently been used in [33,34] as a means to represent local structural information in order to classify symbol layout and in [1] for symbol segmentation.

Given a set of points $\{p_1, p_2, \ldots, p_n\}$ on the contour of a shape, shape context is defined for each point p_i as a log-polar histogram of the remaining points relative to p_i. If two shapes are similar, it is expected that the shape context of points located on a same region on both shapes will present a similar shape context. Similarity between shape contexts of two points p_i and q_j pertaining, respectively, to shapes P and Q is defined via the shape context distance, given by the χ-square distance:

$$C_{ij} = \frac{1}{2}\sum_{k=1}^{K} \frac{[hp_i(k) - hq_j(k)]^2}{hp_i(k) + hq_j(k)} \tag{4}$$

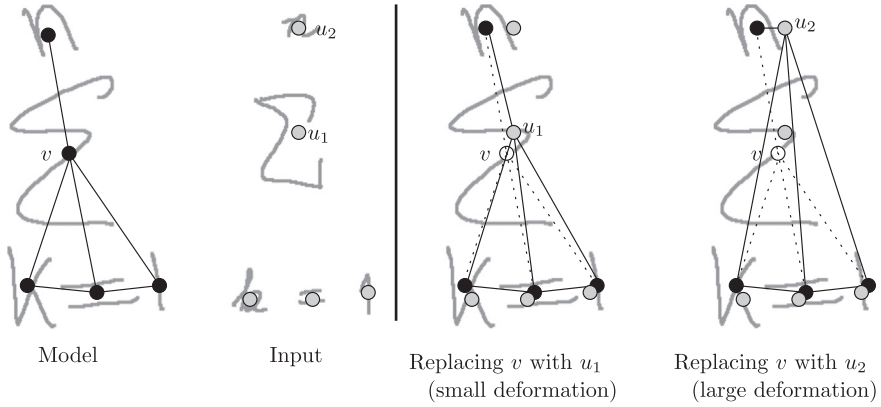

Fig. 2. Model graph structural deformation when one of its vertices, v, is replaced with a vertex of the input graph. Depending on the input graph vertex (u_1 or u_2) that replaces the one in the model graph, deformation can be small or large.

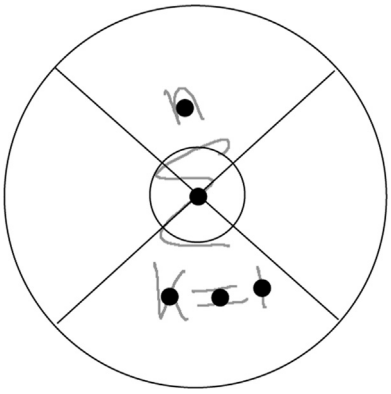

Fig. 3. Expression level shape context bin positioning with respect to symbol $\sum$.

where K is the number of bins, and $hp_i(k)$ and $hq_j(k)$ denote the normalized histogram bins at p_i and q_j, respectively. If both $hp_i(k)$ and $hq_j(k)$ are empty, the term is not considered in the summation.

At expression level, for each vertex (symbol) we compute a shape context considering the coordinates of the remaining vertices of the graph as shown in Fig. 3. The radius of the polar histogram is set as the length of the model graph bounding box diagonal. The matching cost $c_v(global)$ between two vertices can be expressed by the shape context distance given in Eq. (4). Rather than the shape, it captures information on expression structure.

At a symbol level, we first sample n points from the set of strokes of each symbol. Then, for each of these points we compute a shape context taking the remaining $n-1$ points in the same symbol. For each pair of points (one in each symbol), the shape context distance is computed, generating an $n \times n$ cost matrix. An optimum point-to-point matching between points of the two symbols is computed using the Hungarian algorithm [30]. Then, we define $c_v(local)$ as the mean cost of the optimal assignment found (that is, the cost of the optimum assignment divided by n).

The vertex cost is then expressed as a weighted combination of both local and global costs by a parameter γ as follows:

$$c_v = \gamma c_v(local) + (1-\gamma) c_v(global). \quad (5)$$

3. Experimental matching evaluation

We tested the proposed method on two datasets, which are described below. Matching performance was evaluated as described in Section 3.2. In order to determine the optimal values, we varied the parameter values systematically, as detailed in Section 3.3.

3.1. Description of datasets

ExpressMatch dataset consists of transcriptions of 56 model expressions by 25 distinct writers using the *ExpressMatch* software [28], running on a `HP Pavilion tx2500`, with a screen size of 12.1 in and display resolution of 1280×800 pixels. *ExpressMatch* provides a simple and intuitive interface for the users registered as administrators of the system to enter model expressions: each symbol can be manually labeled and ground-truth data (more than one) at expression level can be input textually. So far, only symbol labels and expression in LaTeX format are attached to each model expression. Given a set of model expressions, the system displays them sequentially in a random order and a user can transcribe each of the displayed models. Symbol segmentation is induced during the writing process based on the time gap between strokes, as briefly described in the Introduction and detailed in [28]. The system records user information so that she or he can proceed with the transcription in separate sessions. Each user was asked to write one instance of as many models as their time allowed. Model expressions were taken from Chapters 2 and 4 of [35], without any special criteria.

ExpressMatch dataset contains the collected samples (a total of 901 transcribed expressions) plus the 56 model instances, comprising 81 different types of symbols. The average number of symbols per expression is 22.5, being 4 the minimum and 66 the maximum. Distribution of expression size (number of symbols) and size of each expression class (number of instances) of the ExpressMatch dataset is depicted in the graph of Fig. 4(a).

The second dataset used in this work is the part of the MfrDB dataset [22] that has been made available in the 2013 CROHME edition. We have removed expression classes with only one instance and also four instances that presented error in their ground-truth data. The resulting subset corresponds to a total of 1354 instances in 149 expression classes, comprising 72 distinct types of symbols and an average of 12.13 symbols per expression (see Fig. 4(b)). Notice that function names such as `sin`, `cos`, and `log` are regarded as individual symbols in the MfrDB dataset while in the ExpressMatch dataset each character of these names is individually segmented. Nevertheless, such difference does not affect the applicability of the proposed matching method since no explicit assumption about the set of symbols is made by the method.

Symbol labels in each transcribed expression were initially annotated using a previous version of the matching algorithm and then correspondence was manually verified and corrected, using the ExpressMatch software, to allow automated evaluation of the method.

3.2. Matching evaluation method

Given a model expression and a number of its transcriptions, one could define the mean symbol assignment rate as the average number of symbols in the transcribed expressions that are correctly assigned to the corresponding symbols in the model expression. However, since we have no understanding regarding how a particular model may influence the matching performance, rather than fixing model expressions we propose an evaluation based on an all versus all cross-matching scheme. The main idea in the cross-matching scheme is to make each instance play the model role once against the rest. We assume that every symbol in each instance is uniquely identified, and corresponding symbols in distinct instances have the same identity.

Let C denote a set of instances of an expression with k_C symbols, called an *expression class* or simply *class*, and let $|C|$ denote the number of instances in C. In the cross-matching procedure, we consider matching pairs $(E_m, E_i) \in C \times C$, $i \neq m$, totalizing $|C|(|C|-1)$ matchings.

Given two instances $E_m, E_i \in C$, let $T_C(m,i)$ $(0 \leq T_C(m,i) \leq k_C)$ denote the number of correct symbol assignments when instance E_i is matched to model E_m. Then, considering the $|C|-1$ matchings between E_m and each of the remaining instances, the total number of correct symbol assignments relative to E_m is given by $T_C(m) = \sum_{i=1, i\neq m}^{|C|} T_C(m,i)$ (with $0 \leq T_C(m) \leq k_C(|C|-1)$). The [0,1]-normalized average number of correct symbol assignments per matching given by

$$t_C(m) = \frac{1}{k_C}\frac{T_C(m)}{(|C|-1)} \tag{6}$$

will be called the *model mean* of E_m.

Letting $T_C = \sum_{m=1}^{|C|} T_C(m)$, the [0,1]-normalized expression *class mean* is defined as

$$t_C = \frac{1}{k_C}\frac{T_C}{|C|(|C|-1)}. \tag{7}$$

This value, between 0 and 1, yields an indication of the mean performance per matching in the cross-matching scheme for class C. A large variance indicates that there are matching pairs performing very well or very poorly, but it yields no information regarding the performance of individual instances in the role as a model.

In order to assess whether a given instance $E_m \in C$ is a good model or not, one may compare model mean $t_C(m)$ to class mean t_C. Large negative deviation of $t_C(m)$ from the mean value t_C indicates that instance E_m has poor performance as a model.

Notice that the definitions above apply to a specific expression class C. In order to measure performance over a group of distinct expressions, we consider the average computed over the class means,

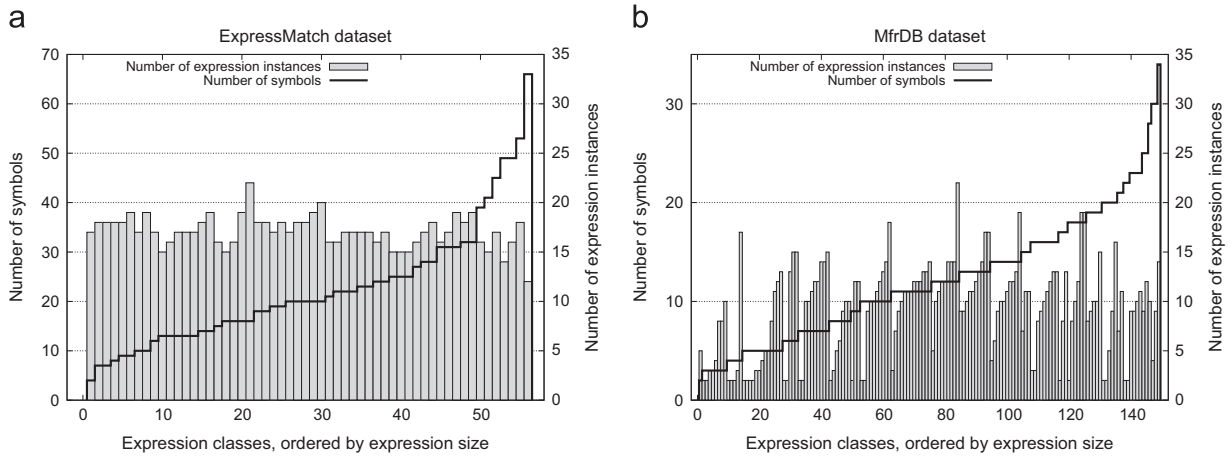

Fig. 4. Sample sets used in the experiments. Left hand axis indicates the number of symbols while right hand axis indicates the number of instances in each expression class. (a) ExpressMatch and (b) MfrDB.

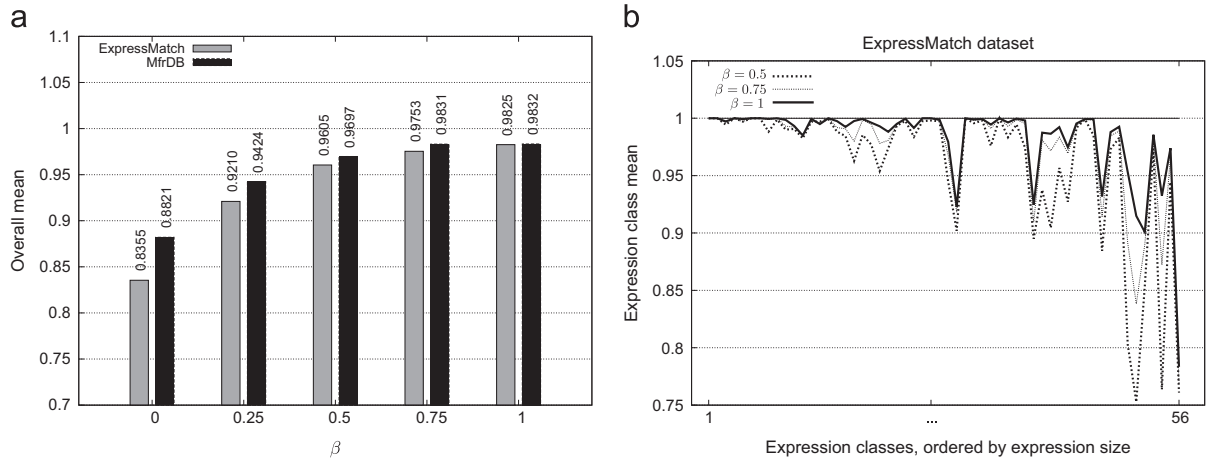

Fig. 5. (a) Overall mean (both datasets) and (b) class mean (ExpressMatch dataset) for different values of β (with α=0 and γ irrelevant, meaning edge costs only). $\beta = 0$ means size component only while β=1 means angular component only.

that is, given n distinct expressions and their respective instance sets (or expression classes) $C_1, C_2, \ldots, C_n$, we define the *overall mean* as

$$t = \frac{1}{n}\sum_{j=1}^{n} t_{C_j}. \tag{8}$$

Classes C_j with class mean t_{C_j} below the overall mean t are those most likely to be the difficult ones.

3.3. Analysis of the cost terms

In order to choose optimal values for parameters β, γ and α, we evaluated matching performance independently on each dataset through the cross-matching scheme (described in Section 3.2) for different combinations of values. To avoid the explosive combination of parameter values, we evaluated edge and vertex costs individually and then we combined the best weighting of both. Specifically, we evaluated the following parameter settings:

- β: weighting between size and angular terms of edge cost (β=0 means "sizes only" and β=1 means "angles only");
- γ: weighting between local and global shape context terms in vertex cost (γ=0 means "global shapes only" and γ=1 means "local shapes only");
- α: weighting between vertex and edge costs; for this, β and γ have been fixed to the optimal values obtained for them (α=0 means "edges only" and α=1 means "vertices only").

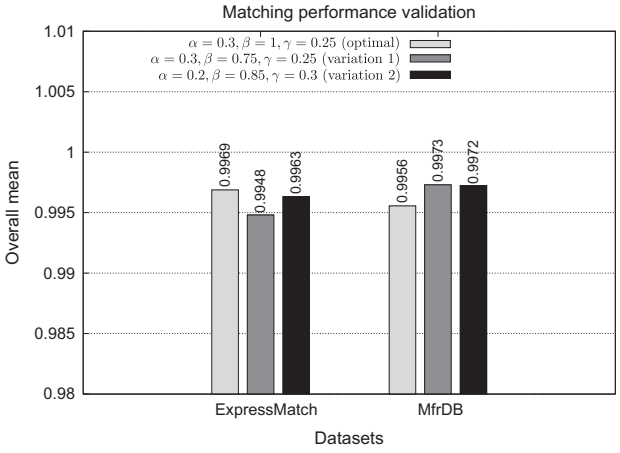

Fig. 8. Parameter validation: for each dataset, the leftmost, middle and rightmost bars correspond to the overall mean obtained, respectively, with the optimal parameters β=1, γ=0.25 and α=0.3, with β=0.75, γ=0.25 and α=0.3, and β=0.85, γ=0.3 and α=0.2 (values close to the optimal ones, but chosen arbitrarily).

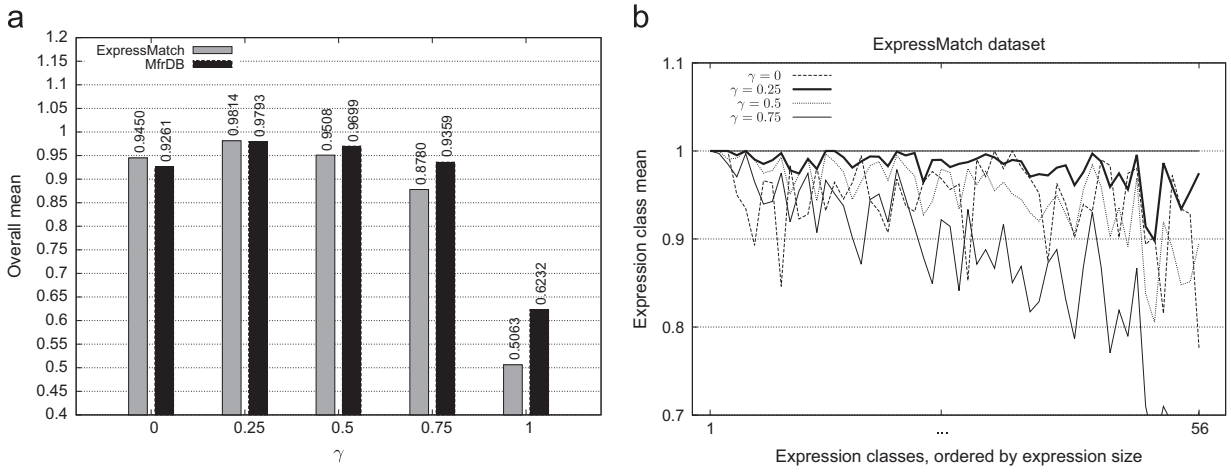

Fig. 6. (a) Overall mean (both datasets) and (b) class mean (ExpressMatch dataset) for different values of γ (with α=1 and β irrelevant, meaning vertex costs only). γ=0 means global shape context only while γ=1 means local shape context only.

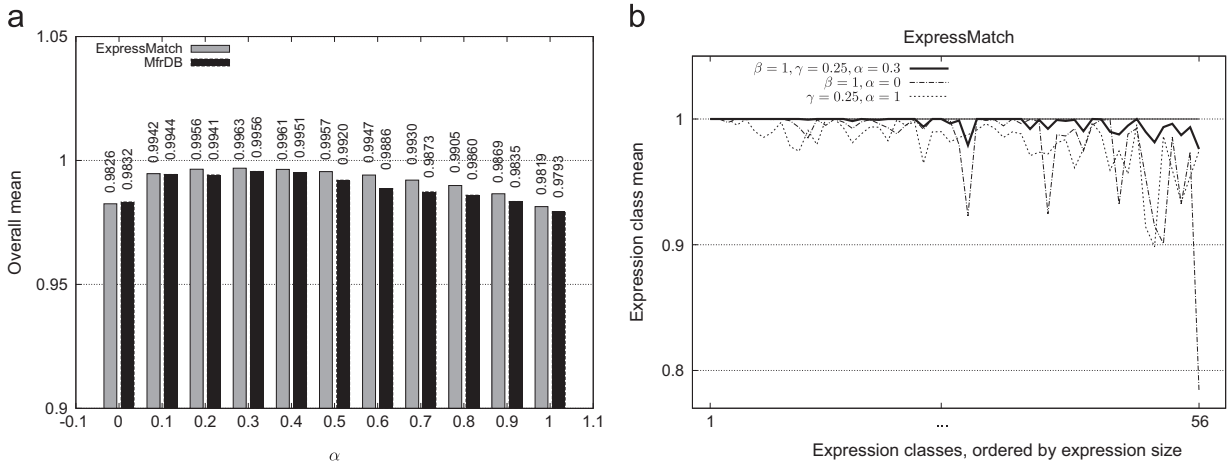

Fig. 7. (a) Overall mean for $\alpha \in \{0, 0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8, 0.9, 1\}$. We use β=1 (edge cost with angular term only) and γ=0.25 (vertex cost with 25% local shape context term and 75% global shape context term). (b) Class mean for the ExpressMatch dataset, considering optimal edge cost (β=1, γ not relevant, α=0), optimal vertex cost (β not relevant, γ=0.25, α=1), and optimal edge-vertex combined cost (β=1, γ=0.25, α=0.3).

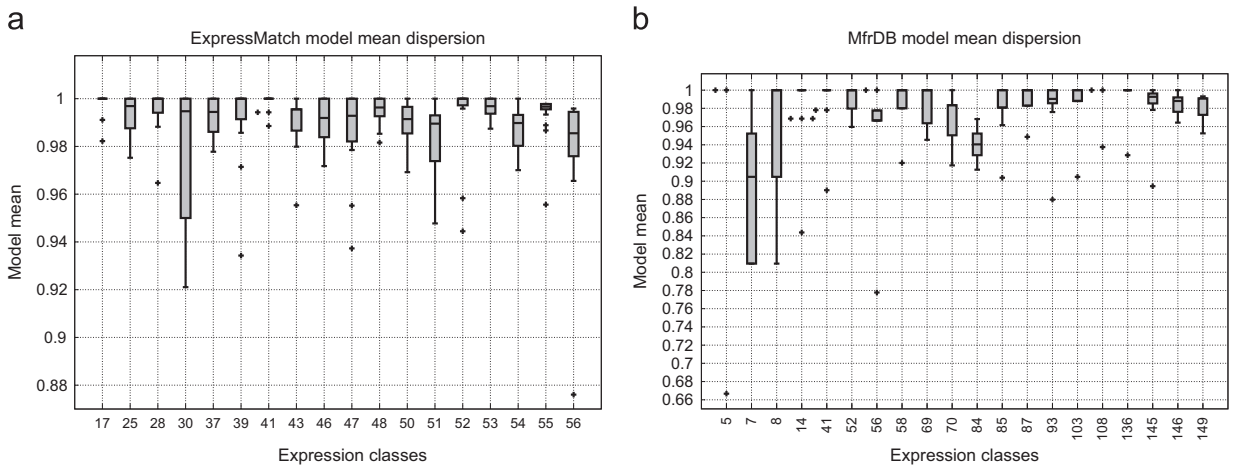

Fig. 9. Model mean dispersion ($\beta=1$, $\gamma=0.25$, $\alpha=0.3$) for expression classes with class mean below 0.99 or with large dispersion: ExpressMatch (left) and MfrDB (right).

3.3.1. Edge cost (parameter β)

Fig. 5(a) shows the overall mean for $\beta \in \{0, 0.25, 0.5, 0.75, 1\}$. We observe that matching performance improves with increasing values of β (increasing emphasis on angular deformation and less emphasis on edge size deformation), reaching best performance at $\beta=1$ on both datasets. These values indicate that angular deformation alone (and no size deformation) can account for a good matching performance when only edge cost is considered. Class mean, relative to each of the expression classes of ExpressMatch dataset, for some values of β, is shown in Fig. 5(b).

3.3.2. Vertex cost (parameter γ)

Vertex cost combines local and global shape contexts. For the global shape context, we chose histograms with four angular (determined by the two diagonals) and two radial regions, totalizing eight bins (see Fig. 3); for the local shape context, we applied to each symbol a pre-processing consisting of removing duplicated points, eliminating hooks, and smoothing data as described in [36], and then considered a total of $n=30$ points uniformly distributed over the strokes of each symbol and 10 angular and 4 radial regions, totalizing 40 bins. Radial cuts were placed on $r/2^i$, $i=1,2,\ldots,q-1$, where r is the radius and q is the number of radial regions. The number of bins for the global shape context has been chosen based on the empirical evaluation. Very similar results have been obtained also with four angular regions and just one radial region. For the local shape context, the choice of bins was based on results reported in the literature [37]. Matching performance for different values of γ is shown in Fig. 6.

For $\gamma=0.25$, an overall good performance regardless of expression size is obtained, for both datasets. For larger values of γ (more emphasis on local shape context), we observe a small inverse correlation between matching performance and expression size (see Fig. 6(b)). These results show that global shape context alone ($\gamma=0$) can capture most of the structural information and that it is sufficient for a good matching rate. In conjunction with local shape context it yields better matching performance as can be seen by comparing performance for $\gamma=0$ (only global shape context) and $\gamma=0.25$ (25% of local shape context and 75% of global shape context).

3.3.3. Combined cost (parameter α)

Based on the results described above, we fixed the values of parameter β for the edge cost and parameter γ for the vertex cost to be the best values found, that is, $\beta=1$ and $\gamma=0.25$. Matching performance for different values of α, using these fixed values for β and γ, is shown in Fig. 7(a).

Although matching rates are already very high for individual cost terms – edges only ($\alpha=0$) or vertices only ($\alpha=1$), and only little room is left for improvement, the combination provides an overall improvement, specially for α around 0.1–0.5. To evince the improvement achieved when vertex and edge costs are combined, we show in Fig. 7(b) a comparison of matching performance over the ExpressMatch dataset for $\alpha \in \{0, 0.3, 1\}$. Combination benefit is clear over the entire set, being more noticeable for expressions with larger number of symbols.

Since classes have different numbers of instances and expressions have different numbers of symbols, the [0,1]-normalized mean values do not help one to estimate the proportion of symbols matched incorrectly or the proportion of expressions matched incorrectly, overall. Considering the cross-matching scenario and the chosen parameter values, for the ExpressMatch dataset $\frac{522}{334,898} \approx 0.16\%$ of symbol associations were incorrect and $\frac{200}{15,540} \approx 1.29\%$ matchings presented error; for the MfrDB dataset, $\frac{971}{182,782} \approx 0.5\%$ of symbol associations were incorrect and $\frac{176}{14,184} \approx 1.2\%$ of matchings presented error.

3.4. Parameter validation

The computed optimal parameter values are the same for both datasets. Since $\beta=0.75$ and $\beta=1$ presented very close results on MfrDB dataset, we also processed both datasets with $\beta=0.75$. In addition, both datasets were also processed using parameter values $\beta=0.85$, $\gamma=0.3$ and $\alpha=0.2$, values close to the optimal ones but chosen arbitrarily. As can be seen in Fig. 8, when parameter values are slightly changed around the optimal ones, the matching performance suffers a marginal decrease on ExpressMatch dataset while an opposite effect is observed with respect to the MfrDB dataset.

Since the choice of parameter values was not based on an exhaustive search, it is expected that the chosen values may not be optimum.

4. Matching analysis

In this section we examine classes that presented poor class mean in order to diagnose where matching is failing.

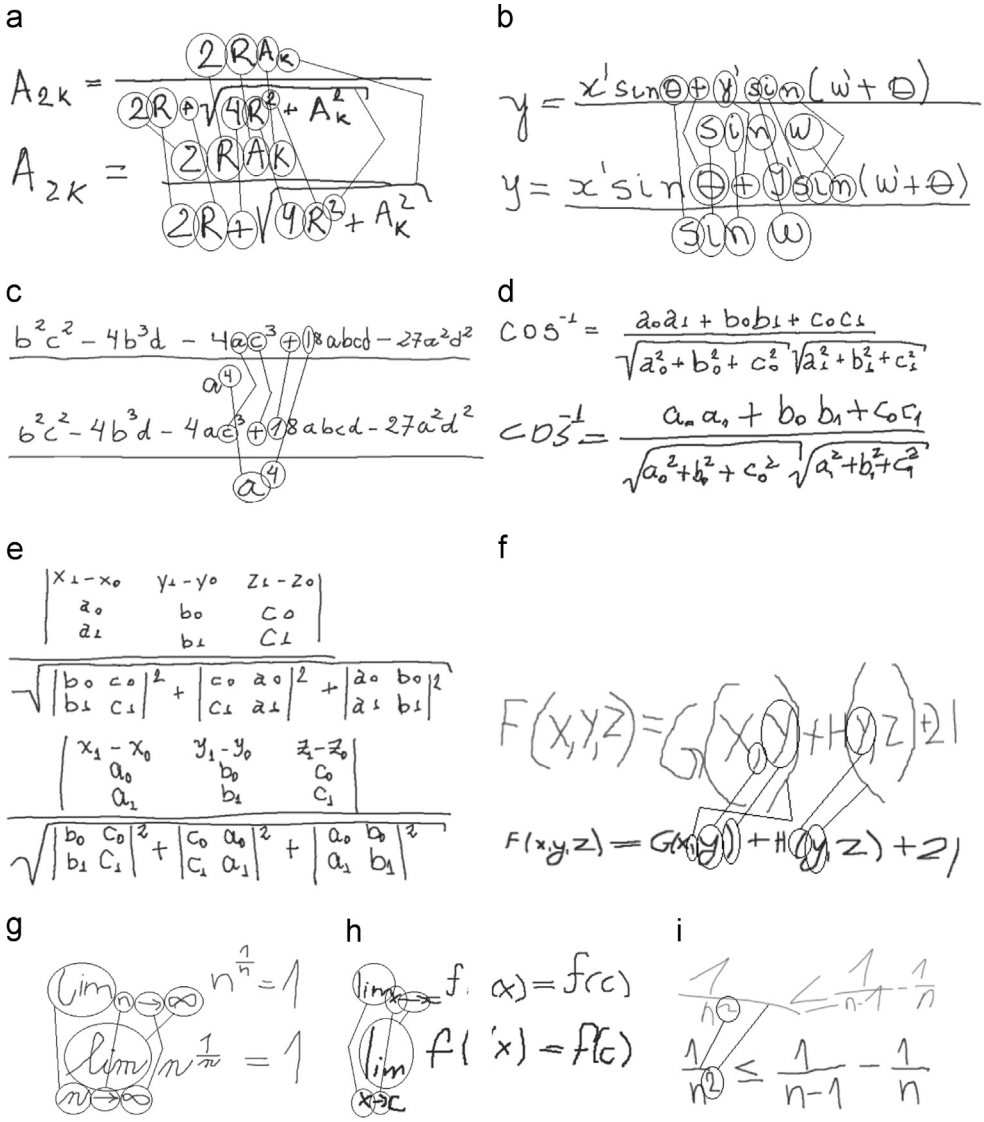

Fig. 10. Outliers of some classes of both datasets, together with a worst matching pair. Matching errors are explicitly indicated in some pairs. (a) ExpressMatch 30, (b) ExpressMatch 39, (c) ExpressMatch 47, (d) ExpressMatch 52, (e) ExpressMatch 56, (f) MfrDB 145, (g) MfrDB 56, (h) MfrDB 93, and (i) MfrDB 103.

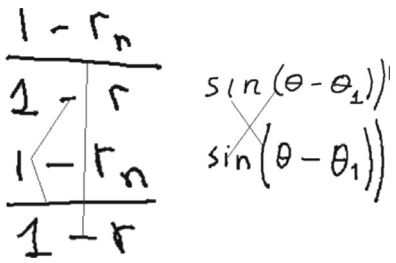

Fig. 11. Assignment error due to confusion between – (minus) and —— (fraction line) and between symbols "(" and "*i*".

4.1. Model mean dispersion

In the ExpressMatch and MfrDB datasets, 30 out of 56 and 110 out of 149 expression classes, respectively, presented no matching errors. The dispersion of model mean (see Eq. (6)) of expression classes with class mean below 0.99 or with large dispersion is shown in Fig. 9.

Notice that the number of instances and symbols vary from class to class and since the mean values are [0,1]-normalized, they cannot be directly compared across the classes (for example, for a model with four symbols, two matching errors imply a 50% error, while for a model with 20 symbols, two errors imply a 10% error). Nevertheless, for each class, those points far below the class mean correspond to instances with poor performance as models. In some classes there are outlier instances that perform much poorly than the average, while in other classes dispersion is just around the class mean. If class mean is computed disregarding the mean of the worst performing model instances in each class, there is a non-negligible gain in overall mean. For instance, by disregarding one, two and three worst model instances of each class, the overall mean of ExpressMatch dataset goes from 0.9963 to 0.9976, 0.9981 and 0.9983. In the MfrDB dataset,[1] it goes from 0.9956 to 0.9976, 0.9982, and 0.9985.

4.2. Common matching errors

Understanding symbol assignment errors may provide important information to improve the method as well as to guide the

[1] Worst performing model instance means are disregarded only if there is still at least one instance left in the class.

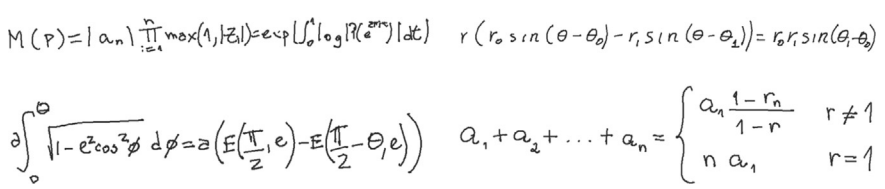

Fig. 12. Samples that presented matching error due to confusion between similar symbols.

Table 1
Assignment errors due to the confusion between similar symbols can be significantly reduced by reducing weight of local shape context.

ExpressMatch (expression class)	Class mean (varying γ)		
	$\gamma=0.05$	$\gamma=0.10$	$\gamma=0.25$
(51) $r(r_0 \sin(\theta-\theta_0)-r_1 \sin(\theta-\theta_1))=r_0 r_1 \sin(\theta_1-\theta_0)$	0.9962	0.9923	0.9814
(54) $M(P)=\lvert a_n\rvert\prod_{i=1}^{n}\max(1,\lvert z_i\rvert)=\exp\left(\int_0^1 \log\lvert P(e^{2\pi i t})\rvert\, dt\right)$	0.9926	0.9934	0.9871
(50) $a\int_0^{\theta}\sqrt{1-e^2\cos^2\phi}\, d\phi = a\left(E\left(\frac{\pi}{2},e\right)-E\left(\frac{\pi}{2}-\theta,e\right)\right)$	0.9972	0.9968	0.9897
(46) $a_1+a_2+\cdots+a_n=\begin{cases} a_1\frac{1-r_n}{1-r} & r\neq 1 \\ na_1 & r=1 \end{cases}$	1	0.9998	0.9898

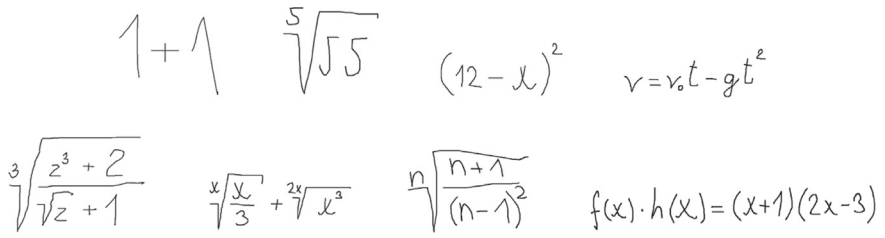

Fig. 13. Some simple expressions with assignment errors that are due to the unstable nature of global shape context.

choice of model instances. Fig. 10 shows some of the outliers in the both datasets, together with a respective worst matching pair in the same class.

We have observed that a common assignment error is caused by the presence of similar symbols close each other. For instance, confusion between – (minus) and —— (fraction line), or between "(" and "*i*", or between two ")" as the ones shown in Fig. 11 were observed in several expressions.

Fig. 12 shows additional examples of expressions where significant error due to confusion between similar symbols was observed.

Most of these types of errors are corrected by reducing the weight of local shape context (small value for γ), as can be seen in Table 1.

Another common type of error seems to be related to the unstable nature of global shape context. Such unstability may occur in simple expressions like the ones shown in Fig. 13, either due to the reduced number of symbols (points) used for the computation of shape context or because the relative position between symbols (represented by their centroid) is not properly conveyed (for instance, cases that involve symbol $\sqrt{}$ or presence of arrangements such as "(*y*" as in the expression of Fig. 10(f)). For these expressions, increasing the weight of local shape context (i.e., larger values for γ) seems to greatly reduce the errors (see Table 2). Equivalently, using α close to zero (i.e., no vertex cost and thus no shape context) also produces similar effects.

However, if a same symbol is written in a too different way, a large value of γ may worsen the result. For instance, for expression $t=t_0/\sqrt{(1-(v^2/c^2))}$, for γ equal to 0, 0.25 and 0.40 the class means are, respectively, 0.9872, 0.9765 and 0.9658, showing that increasing γ does not help. This is because there are several symbols that are written too differently from one instance to another as shown in Fig. 14.

Another reason for the unstable behavior of global shape context is the displacement variation among substructures, such as between numerators and denominators (see, for instance, the first five examples of Fig. 10). For these cases, increasing γ (i.e., more emphasis on symbol shape context and less on global shape context) significantly improves results, as shown in Table 3.

4.3. Discussions

In our implementation, symbol rotation and writing variation are not given special consideration. Thus, corresponding symbols may present a large shape context distance. Nevertheless, this is not critical to establish a good matching because it is likely that shape context distance to non-corresponding symbols in the expression is equally large. Further, for particular expressions, it is clear from the analysis in the previous section that symbol cost plays an important role in reducing assignment errors.

Thus, improving symbol similarity cost should lead to an overall improvement. For instance, matching cost computation may take into consideration the fact that some symbols can have different shapes, depending on who writes them. A set of representative variations of a same symbol could be kept and the optimal matching formulation could consider all the variations of a given symbol, choosing the most similar one. Furthermore, as vertex costs should express dissimilarity between the symbols being matched, any symbol features and symbol similarity measures could be included in the matching cost. Some local structural information could also be considered in order to avoid confusions between similar symbols (e.g., between a minus signal and a fraction line).

Structural information of short expressions or of those that involve sub-expressions whose relative position may vary significantly from instance to instance is particularly affected by the unstability of global shape context. In most of these cases, emphasis on symbol shape context seems to correct the distortion (unless there are too many similar symbols in the expression).

Experimental results indicate that, at least when using complete graphs, the parameter β used to weight size and angular information of edges (Eq. (2)) could be removed and the cost function be defined only in terms of angular deformation.

Table 2
Examples of expressions for which emphasis on local shape context (symbol features) helps in establishing the right correspondence.

MfrDB (expression class)	Class mean (varying γ)			
	$\gamma=0.05$	$\gamma=0.25$	$\gamma=0.40$	$\gamma=0.80$
(7) $1+1$	0.8810	0.8929	0.9286	0.9762
(14) $\sqrt[5]{5}$	0.9614	0.9853	0.9926	1.0000
(41) $(12-x)^2$	0.9623	0.9890	0.9984	1.0000
(52) $v=v_0t-gt^2$	0.9714	0.9899	0.9966	0.9966
(56) $\lim_{n\to\infty}n^{1/n}=1$	0.9444	0.9567	0.9578	0.9744
(93) $\lim_{x\to c}f(x)=f(c)$	0.9663	0.9859	0.9926	0.9980
(136) $f(x)\cdot h(x)=(x+1)(2x-3)$	0.9773	0.9898	0.9966	0.9977
(145) $F(x,y,z)=G(x,y)+H(y,z)+21$	0.9342	0.9839	0.9876	0.9897
(70) $\sqrt[3]{\frac{z^3+2}{\sqrt{z}+1}}$	0.9298	0.9683	0.9835	0.9986
(69) $\sqrt[x]{\frac{x}{3}}+\sqrt[2y]{x^3}$	0.9785	0.9884	0.9950	1.0000
(84) $\sqrt[n]{\frac{n+1}{(n-1)^3}}$	0.9127	0.9394	0.9527	0.9769

Fig. 14. Writing variation of a same symbol: from top to bottom, symbols “2”, “v” and “t”.

Table 3
Examples of expressions for which emphasis on local shape context (symbol features) helps in establishing the right correspondence.

ExpressMatch (expression class)	Class mean (varying γ)				
	$\gamma=0.05$	$\gamma=0.10$	$\gamma=0.25$	$\gamma=0.35$	$\gamma=0.50$
(30) $A_{2k}=\frac{2RA_k}{2R+\sqrt{4R^2+A_k^2}}$	0.9487	0.9604	0.9791	0.9875	0.9939
(47) $\frac{b^2c^2-4b^3d-4ac^3+18abcd-27a^2d^2}{a^4}$	0.9582	0.9724	0.9876	0.9909	0.9925
(56) $\left\|\begin{vmatrix} x_1-x_0 & y_1-y_0 & z_1-z_0 \\ a_0 & b_0 & c_0 \\ a_1 & b_1 & c_1 \end{vmatrix}\right\| \Big/ \sqrt{\left\|\begin{vmatrix} b_0 & c_0 \\ b_1 & c_1 \end{vmatrix}^2+\begin{vmatrix} c_0 & a_0 \\ c_1 & a_1 \end{vmatrix}^2+\begin{vmatrix} a_0 & b_0 \\ a_1 & b_1 \end{vmatrix}^2\right\|}$	0.8892	0.9358	0.9761	0.9783	0.9814
(39) $y=\frac{x'\sin\theta+y'\sin(\omega'+\theta)}{\sin\omega}$	0.9491	0.9701	0.9922	0.9964	0.9973
(52) $\cos^{-1}=\frac{a_0a_1+b_0b_1+c_0c_1}{\sqrt{a_0^2+b_0^2+c_0^2}\sqrt{a_1^2+b_1^2+c_1^2}}$	0.9475	0.9636	0.9935	0.9968	0.9953

Nevertheless, we keep it since it may be useful when non-necessarily complete graphs are considered.

Some of the observed common errors suggest that neatly written expressions (equally spaced symbols, symbols with adequate proportional sizes, and correctly aligned sub-expressions, for instance, with respect to centralization, and with no rotation) tend to be good models, being consonant with common sense.

When multiple models are available for a same expression, a cross-matching simulation can be run, and those that have poor performance can be removed from the set of models. Additionally, an input expression could be matched to each of the models and a cost function that combines costs relative to the several models could be considered.

As discussed in Section 3.4, slight changes of parameter values around the optimal ones do not affect much the overall matching performance. Rather than trying to adjust parameters exhaustively, which may lead to overfitting, or considering a quadratic assignment problem formulation (for instance, as an exact graph matching problem [38]) which is more difficult, we believe that improved matching rates could be achieved by simply improving matching cost expressiveness as discussed above.

5. Conclusions and future work

We have proposed an expression matching approach for automatically labeling the different components of a transcribed expression with ground-truth data. Experimental results on two large independent datasets show that the proposed approach, based on a naive modeling of the problem as a simple linear assignment problem, provides surprisingly good matching results.

Matching costs defined as a weighted sum of very general edge and symbol features have shown to be expressive. In particular, cost terms that capture structural information are sufficient for good matching rates, and symbol feature helps to slightly improve the rate. An overall mean symbol assignment rate above 99% is achieved with the proposed algorithm. Some common matching errors were identified and we observed that most of them can be properly corrected by adjusting the weight between local and global shape contexts used in the vertex cost. Both *ExpressMatch* (the system used to collect data and perform matching evaluation) and the generated dataset, annotated with ground-truth data at symbol and expression levels, are publicly available from `http://escience.ime.usp.br/projects/expressmath/`. Some possible improvements regarding cost function are discussed in Section 4.3.

In the proposed procedure, models are manually created. Thus, there is a need to individually label each of the symbols and to attach ground-truth data to these expressions. Nevertheless, a nice characteristic of the proposed approach is the fact that several ground-truth annotated samples of each model expression can be easily obtained, allowing the creation of an expressive dataset in terms of writing variations. Furthermore, since our method does not impose restrictions on symbol classes or mathematical notation, it is general enough for the creation of ME datasets with distinct features.

A major challenge that remains is to extend the approach to deal with unconstrained data with no requirements relative to previous segmentation of the symbols. Such extension would facilitate the creation of datasets with ground-truth data for the development and testing of solutions for general pen based devices and also for offline handwriting.

Conflict of interest

None declared.

Acknowledgment

This work is supported by FAPESP (scholarship to F.D.J.A, grants 2012/08389-1 and 2013/13535-0), and by CNPq (grant 560165/2010-2), Brazil. N.S.T.H. is partly supported by CNPq (grant 307357/2012-0). The authors thank Dr. A. Noma, who gently provided part of the matching code, W.Y. Honda who participated in an early stage of this research, all volunteers who took their time to transcribe the expressions, and the reviewers who provided valuable comments that greatly helped to improve this paper.

References

[1] L. Hu, R. Zanibbi, Segmenting handwritten math symbols using adaboost and multi-scale shape context features, in: Twelfth International Conference on Document Analysis and Recognition (ICDAR), 2013, pp. 1180–1184.
[2] U. Garain, B.B. Chaudhuri, Recognition of online handwritten mathematical expressions, IEEE Trans. Syst. Man Cybern. Part B 34 (6) (2004) 2366–2376.
[3] S.M. Watt, X. Xie, Recognition for large sets of handwritten mathematical symbols, in: Proceedings of the International Conference on Document Analysis and Recognition, vol. 2, 2005, pp. 740–744.
[4] H.-J. Winkler, M. Lang, Symbol segmentation and recognition for understanding handwritten mathematical expressions, in: A. Downton, S. Impedovo (Eds.), Progress in Handwriting Recognition, World Scientific, Singapore, 1997, pp. 407–412.
[5] B. Huang, M.-T. Kechadi, A structural analysis approach for online handwritten mathematical expressions, Int. J. Comput. Sci. Netw. Secur. 7 (7) (2007) 47–56.
[6] R. Genoe, J. Fitzgerald, T. Kechadi, An online fuzzy approach to the structural analysis of handwritten mathematical expressions, in: IEEE International Conference on Fuzzy Systems, 2006, pp. 244–250.
[7] L. Zhang, D. Blostein, R. Zanibbi, Using fuzzy logic to analyze superscript and subscript relations in handwritten mathematical expressions, in: Proceedings of the Eighth International Conference on Document Analysis and Recognition, 2005, pp. 972–976.
[8] A.-M. Awal, H. Mouchère, C. Viard-Gaudin, A global learning approach for an online handwritten mathematical expression recognition system, Pattern Recognit. Lett. 35 (2014) 68–77.
[9] A.-M. Awal, H. Mouchère, C. Viard-Gaudin, Improving online handwritten mathematical expressions recognition with contextual modeling, in: Proceedings of the 12th International Conference on Frontiers in Handwriting Recognition, 2010, pp. 427–432.
[10] T.H. Rhee, J.H. Kim, Efficient search strategy in structural analysis for handwritten mathematical expression recognition, Pattern Recognit. 42 (2009) 3192–3201.
[11] A.-M. Awal, H. Mouchère, C. Viard-Gaudin, Towards handwritten mathematical expression recognition, in: Proceedings of the 10th International Conference on Document Analysis and Recognition, 2009, pp. 1046–1050.
[12] Y. Shi, H. Li, F.K. Soong, A unified framework for symbol segmentation and recognition of handwritten mathematical expressions, in: Proceedings of the Ninth International Conference on Document Analysis and Recognition, vol. 02, 2007, pp. 854–858.
[13] S. MacLean, G. Labahn, E. Lank, M. Marzouk, D. Tausky, Grammar-based techniques for creating ground-truthed sketch corpora, Int. J. Doc. Anal. Recognit. 14 (2011) 65–74.
[14] R. Zanibbi, A. Pillay, H. Mouchère, C. Viard-Gaudin, D. Blostein, Stroke-based performance metrics for handwritten mathematical expressions, in: International Conference on Document Analysis and Recognition, 2011, pp. 334–338.
[15] K. Sain, A. Dasgupta, U. Garain, EMERS: a tree matching-based performance evaluation of mathematical expression recognition systems, Int. J. Doc. Anal. Recognit. 14 (2011) 75–85.
[16] A.-M. Awal, H. Mouchère, C. Viard-Gaudin, The problem of handwritten mathematical expression recognition evaluation, in: Proceedings of the 12th International Conference on Frontiers in Handwriting Recognition, 2010, pp. 646–651.
[17] A. Lapointe, D. Blostein, Issues in performance evaluation: a case study of math recognition, in: Proceedings of the 10th International Conference on Document Analysis and Recognition, 2009, pp. 1355–1359.
[18] R. Zanibbi, D. Blostein, Recognition and retrieval of mathematical expressions, Int. J. Doc. Anal. Recognit., 2011, pp. 1–27.
[19] K.-F. Chan, D.-Y. Yeung, Mathematical expression recognition: a survey, Int. J. Doc. Anal. Recognit. 3 (2000) 3–15.
[20] D. Blostein, A. Grbavec, Recognition of mathematical notation, in: H. Bunke, P. Wang (Eds.), Handbook of Character Recognition and Document Image Analysis, World Scientific, Singapore, 1997, pp. 557–582.
[21] S. Quiniou, H. Mouchère, S. Saldarriaga, C. Viard-Gaudin, E. Morin, S. Petitrenaud, S. Medjkoune, HAMEX—a handwritten and audio dataset of mathematical expressions, in: 2011 International Conference on Document Analysis and Recognition (ICDAR), 2011, pp. 452–456.
[22] J. Stria, M. Bresler, D. Prusa, V. Hlavác, MfrDB: database of annotated on-line mathematical formulae, in: 2012 International Conference on Frontiers in Handwriting Recognition (ICFHR), 2012, pp. 542–547.

[23] H. Mouchère, C. Viard-Gaudin, D.H. Kim, J.H. Kim, U. Garain, CROHME2011: Competition on recognition of online handwritten mathematical expressions, in: International Conference on Document Analysis and Recognition, 2011, pp. 1497–1500.
[24] H. Mouchère, C. Viard-Gaudin, D. Kim, J. Kim, U. Garain, ICFHR 2012 competition on recognition of on-line mathematical expressions (CROHME), in: International Conference on Frontiers in Handwriting Recognition, 2012, pp. 811–816.
[25] H. Mouchère, C. Viard-Gaudin, R. Zanibbi, U. Garain, D. H. Kim, J.H. Kim, ICDAR 2013 CROHME—Third international competition on recognition of online mathematical expressions, in: Proceedings of the International Conference on Document Analysis and Recognition, 2013, pp. 1428–1432.
[26] A. Lapointe, Issues in performance evaluation of mathematical notation recognition systems (Master's thesis), Queen's University, 2008.
[27] N.S.T. Hirata, W.Y. Honda, Automatic labeling of handwritten mathematical symbols via expression matching, in: Proceedings of the Eighth International Conference on Graph-based Representations in Pattern Recognition, 2011, pp. 295–304.
[28] F.D.J. Aguilar, N.S.T. Hirata, ExpressMatch: a system for creating ground-truthed datasets of online mathematical expressions, in: Proceedings of 10th IAPR International Workshop on Document Analysis Systems, 2012, pp. 155–159.
[29] The ExpressMatch Website, ⟨http://code.google.com/p/express-match/⟩, 2012.
[30] H.W. Kuhn, The Hungarian method for the assignment problem, Naval Res. Logist. Quart. 2 (1955) 83–97.
[31] A. Noma, A. Pardo, R.M. Cesar Jr., Structural matching of 2D electrophoresis gels using deformed graphs, Pattern Recognit. Lett. 32 (2011) 3–11.
[32] S. Belongie, J. Malik, J. Puzicha, Shape matching and object recognition using shape contexts, IEEE Trans. Pattern Anal. Mach. Intell. 24 (2002) 509–522.
[33] L. Ouyang, A symbol layout classification for mathematical formulas using layout context (Master's thesis), Rochester Institute of Technology, 2009.
[34] F. Alvaro, R. Zanibbi, A shape-based layout descriptor for classifying spatial relationships in handwritten math, in: Proceedings of the 2013 ACM Symposium on Document Engineering, 2013, pp. 123–126.
[35] D. Zwillinger, Standard Mathematical Tables and Formulae, 31st ed., Chapman and Hall, CRC, London, 2003.
[36] B.Q. Huang, Y.B. Zhang, M.T. Kechadi, Preprocessing techniques for online handwriting recognition, in: Proceedings of the Seventh International Conference on Intelligent Systems Design and Applications, 2007, pp. 793–800.
[37] L. Prasanth, V.J. Babu, R.R. Sharma, G.V.P. Rao, M. Dinesh, Elastic matching of online handwritten Tamil and Telugu scripts using local features, in: Proceedings of the Ninth International Conference on Document Analysis and Recognition, vol. 02, 2007, pp. 1028–1032.
[38] D. Conte, P. Foggia, C. Sansone, M. Vento, Thirty years of graph matching in pattern recognition, in: Int. J. Pattern Recognit. Artif. Intell. (2004) 265–298.

Nina S.T. Hirata holds a PhD degree in Computer Science from University of São Paulo, Brazil. Currently, she is an associate professor of the Computer Science Department, Institute of Mathematics and Statistics, at the same university. Her research interests include machine learning, image segmentation and analysis, morphological operator learning, and handwriting recognition.

Frank D. Julca-Aguilar received the BSc degree in Computer Science from National University of Trujillo (UNT), Peru. He is a graduate student in Computer Science at University of São Paulo, Brazil. His current research interests include machine learning, handwriting recognition, speech recognition and content-based image retrieval.