2012 10th IAPR International Workshop on Document Analysis Systems

ExpressMatch : a system for creating ground-truthed datasets of online mathematical expressions

Frank D. J. Aguilar and Nina S. T. Hirata
Department of Computer Science
Institute of Mathematics and Statistics / University of São Paulo
São Paulo, Brazil
Email: faguilar@ime.usp.br, nina@ime.usp.br

***Abstract*—In recognition domains, publicly available ground-truthed datasets are essential to perform effective performance evaluation and comparison of existing methods and systems. However, in the field of online handwritten mathematical expression recognition, datasets are quite scarce and their creation is one of the current challenging issues. In this paper, we present *ExpressMatch*, a system designed to help creation and management of online mathematical expression datasets with ground-truth data. In this system, handwritten model expressions can be input and manually annotated with ground-truth data; transcriptions of these expressions can be automatically annotated by matching them to the respective models. Additional metadata can also be attached to each sample expression. To test the system, a dataset consisting of 56 model expressions and 910 sample expressions with a total of 20,010 symbols, written by 25 different writers, has been created. This dataset, as well as *ExpressMatch*, will be made publicly available.**

***Keywords*-online mathematical expressions; ground-truthed dataset; performance evaluation.**

I. Introduction

Devices such as tablets, hand-held PDAs, and electronic whiteboards have emerged and become very popular. They provide a more effective and natural way to input non-usual entries, such as diagrams and equations, into computer systems. To take full advantage of such features, handwriting recognition is crucial. Pen based devices produce online data, in which timing information about the writing process, such as the order and velocity in which strokes are written, is available.

Mathematical expression (ME) recognition figures as one of the current challenging problems in the field of handwriting recognition. Many technical documents include some mathematical formula and their input is usually performed with a special typesetting command such as LaTeX or by using mechanisms such as symbol selection tools. Availability of ME recognition systems would allow users to enter mathematical expressions naturally, in a similar way they are used to hand write them on a sheet of paper.

The recognition process of MEs can be roughly divided into three steps: (i) symbol segmentation, (ii) symbol classification, and (iii) structural analysis and interpretation. The segmentation step consists in grouping strokes belonging to a same symbol. The symbol recognition step consists in associating a label to each segmented symbol. In the last step, an internal hierarchical structure is used to represent spatial and logical relations among symbols, and finally that structure is processed to generate a result (e.g. a LaTeX representation of the input expression).

It is generally acknowledged that publicly available datasets with ground-truth data are essential when evaluating performance of existing methods or systems: weakness and strengths of different methods/systems can be determined by testing them on a common dataset [1]. Public datasets allow reproduction of experiments to validate or negate the results [2]. Indeed, their accessibility enables contests which have proven useful for many fields [3].

However, in the domain of recognition of handwritten MEs, publicly available datasets are quite scarce [2]. Systems developed in this domain have been mostly evaluated and tested on individually collected datasets (e.g. [4], [5], [6], [7]). The non-existence of large and expressive online ME datasets makes an effective performance evaluation and comparison of available systems a difficult task.

Unfortunately, the process of creating datasets with ground-truth data is labor-intensive and error-prone [8]. A large and expressive dataset should comprise a large variety and number of sample expressions, and ground-truthing them implies the need to label thousands of symbols as well as their relationships individually. To avoid manual labeling of thousands of individual symbols in the sample expressions, part of the dataset creation process should be automated. A possible approach is to consider a set of model expressions annotated with ground-truth data and then automatically annotate samples that are obtained by transcribing the models. Such approach is used, for instance, in [3], [9].

In this work, we propose a Java based system, called *ExpressMatch*, for supporting the creation of online ME datasets. *ExpressMatch* provides functional and practical tools for collecting and organizing data. Given a set of predefined model expressions with ground-truth data, it automatically labels symbols and structure in user transcribed

978-0-7695-4661-2/12 $26.00 © 2012 IEEE
DOI 10.1109/DAS.2012.38

MEs, using the matching approach proposed in [9]. Symbols are segmented during writing time, simplifying the matching task. In addition, the import/export functionality allows sharing and combining data, making possible the creation of large datasets.

The rest of the paper is organized as follows. Section II describes important desired qualities and peculiarities of ME datasets. The main features and the architecture of *ExpressMatch* are described in Section III. Section IV discusses evaluation of the system. Finally, Section V summarizes the main contributions and lists some issues to be further investigated in the context of this work.

II. Desired qualities of ME datasets

Several important qualities that should be fulfilled by testing datasets in order to allow effective performance evaluation of ME recognition methods are pointed in the literature [2], [10], [11]:

- **different levels of labeling** [11]: datasets should have labelings at stroke, symbol and expression levels, in order to allow evaluation of different faces of systems (in [11] it was also proposed labeling at relation level). For example, to evaluate symbol recognition rate (number of correctly recognized symbols over the total number of symbols), ground-truth of each symbol in the expression is needed, while for evaluating segmentation techniques, labeling at stroke level is needed;
- **multiple ground-truths at expression level** [2], [10]: in some cases there are several equivalent ways to represent a given ME within a computer mathematical format. For example, considering LaTeX, x_0^2 can be represented as "x^2_0", "x_0^2","x^{2}_{0}", and "x_{0}^{2}"; all these representations should be accepted as correct;
- **subsets of MEs that meet some constraints**: many systems are designed to work within specific constraints (limited number of symbol classes, limited number of symbols in each ME, specific field of Mathematics, and so on). Thus, MEs should be organized and classified under some established criteria. For each set of constraints, selecting only MEs satisfying the constraints should be possible;
- **statistical representativity** [2], [10]: distribution of ME types within a specific domain should be considered in a dataset, in order to allow a good approximation of the performance of systems in real scenarios;
- **public availability** [2], [10]: evaluation and comparison of different methods on a common dataset would facilitate assessment of weakness and strengths of each system.

III. *ExpressMatch* and dataset creation

ExpressMatch is a system that has been designed to help creation of online ME datasets. The process of creating a dataset starts with the definition of a set of handwritten model expressions. Then, sample expressions are collected by having people transcribing each of the models. The ground-truth information is manually input only for the model expressions; for the transcribed expressions, ground-truth is inherited from their corresponding models. *ExpressMatch* main features are highlighted next:

- the set of model expressions define a corpus. Since models are input manually, the system is very flexible with respect to types of corpora that can be created;
- more than one ground-truth, at expression level, can be attached to each model expression via textual information;
- expressions can be associated to user-defined categories. This feature is useful to select only expressions of a given category;
- user registration and management controls which and how many times a model expression has been transcribed by each `writer`;
- time gap between strokes is taken into consideration to perform segmentation at writing time. Whenever the time gap between two strokes is larger than a given threshold, the system considers that a new symbols is being written. Although this mechanism adds some restriction to the writing style, it has been observed empirically that `writers` have no difficult to adapt themselves to the rule;
- symbols in transcribed expressions are automatically labeled by assigning them to the corresponding symbols in the model expression, based on an expression matching approach proposed in [9];
- matching between symbols in model and transcribed expressions can be visually verified and interactively corrected if necessary;
- both model and transcribed expressions can be added incrementally. In addition, data gathered in different machines can be combined each other. These features make possible an incremental creation of large datasets;
- subsets of expressions can be selected and exported as XML format files. For instance, it is possible to select only expressions of a specific category, or expressions with the number of symbols within a given interval, or expressions written by a specific group of writers;
- it is possible to extract symbol samples in order to create symbol datasets. The set of symbols obtained from the expressions will better resemble the way they are naturally written within MEs than when they are written in a isolated way, and this fact may be relevant for the development of symbol recognizers for ME recognizers.

A. ExpressMatch *architecture*

ExpressMatch consists of six main components, shown in Fig. 1: (1) time-based segmentator (**TBS**) for segmentation of symbols at writing time; (2) model expression capturer (**MC**) for capturing model expressions and annotating them with ground-truth data; (3) instance capturer (**IC**) for capturing transcribed expression instances; (4) expression

matching-based labeler (**EMBL**) for labeling of instance expressions by matching them to the respective model expression; (5) labeling editor (**LE**) for interactive verification and correction of labelings; and (6) importer/exporter of dataset (**IED**) for importing/exporting data.

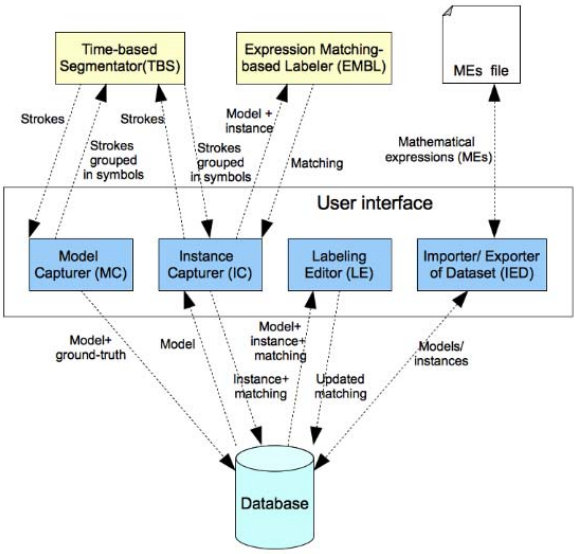


Figure 1. *ExpressMatch* architecture.

Two kinds of users are allowed: `administrators` and `writers`. `Administrators` can define models, evaluate labeling results and use the import/export functionality by interacting respectively with the MC, LE, and IED user interfaces. `Writers` can only write instance expressions by interacting with the IC interface. `Administrators` are also `writers`.

1) Time based segmentator (TBS): given a pair of consecutive strokes, TBS considers them as being part of the same symbol if the temporal gap between the end of the first and the beginning of the second stroke is no longer than a predefined threshold.

2) Model capturer (MC): Model expressions and their corresponding ground-truth data can be input through the MC interface. Figure 2 shows a snapshot of the interface when the model expression $x = \frac{-b \pm \sqrt{b^2-4ac}}{2a}$ is being input. Symbols are segmented during writing: MC sends strokes and their timing information to TBS that returns the strokes grouped as symbols (see Section III-A1). MC indicates which strokes are being considered as belonging to a particular symbol by displaying a bounding box around the set of strokes. The `undo` and `delete` functionality allow correction of possible wrong segmentations. `Undo` eliminates the last written stroke, while `delete` removes an entire symbol (any set of strokes related to a bounding box).

To help organization of MEs, expression classes can be defined and each model expression can be assigned to any

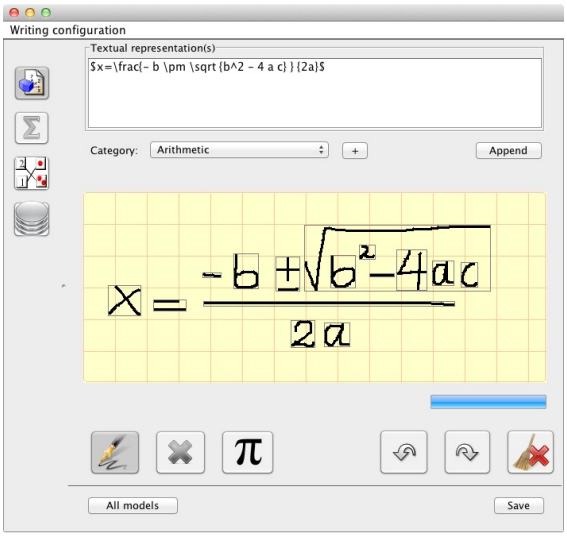


Figure 2. Model collector: expression $x = \frac{-b \pm \sqrt{b^2-4ac}}{2a}$ is being defined as a model.

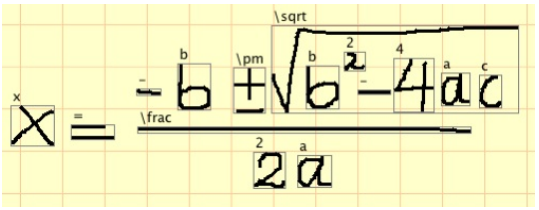


Figure 3. Model expression of Figure 2 with all its symbols labeled.

of those classes. Figure 2 shows that the written model is being assigned to `Arithmetic` category.

Ground-truth data at expression level can be input as textual information through the text area above the written expression. Figure 2 shows ground-truth data in LaTeX format. Additional ground-truth can be assigned using the `append` button, placed below the text area. At symbol level, the π button allows assignment of ground-truth data: labels for each of the symbols can be manually entered, being subsequently shown in the superior corner of each symbol, as shown in Figure 3. Automatic labeling of symbols of model expressions from expression level ground-truth data is an issue for future investigation.

3) Instance capturer (IC): is the interface used to capture instances of model expressions. Model expressions are randomly selected from the set of predefined model expressions and displayed in the superior part of the interface. The system controls which expressions have already been transcribed by each registered user. Figure 4 shows the interface displaying the model expression $cos\theta = \frac{x}{\sqrt{x^2+y^2}}$ and its transcription below it. As the MC component, IC also interacts with the TBS component to get the symbols segmented at writing time. Segmentation is indicated with a bounding box around each symbol and can be corrected

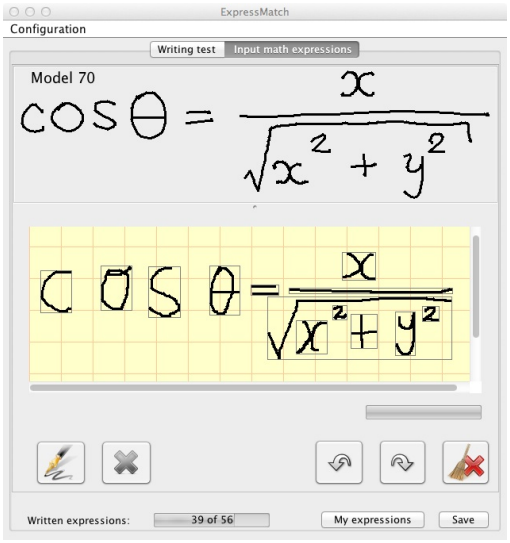


Figure 4. Instance capturer: model expression is shown at the top, to indicate users what instance they have to write.

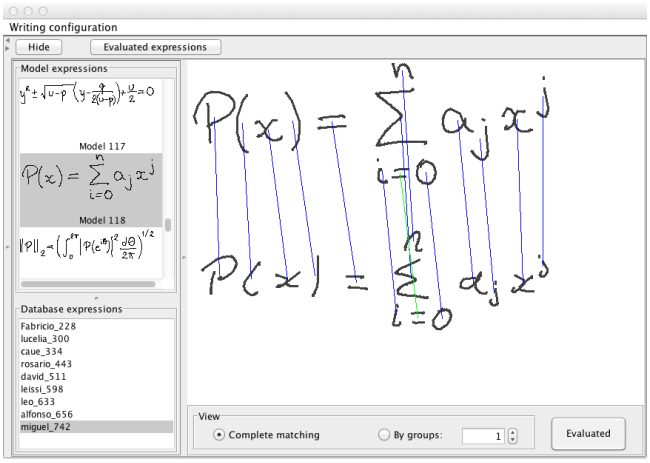


Figure 5. Labeling editor: labeling result is shown as a matching between labeled symbols of model expression (at the top) and unlabeled symbols of the instance expression (at the bottom).

with `undo` and `delete` buttons in a similar way to MC interface. The system issues a warning message if the number of symbols in the model and transcribed expressions differs.

4) Expression matching based labeler (EMBL): symbols of an instance expression are labeled automatically, based on the method described in [9]. Each time an instance expression is input, EMBL computes a matching between that instance and its corresponding model, finding a one-to-one correspondence between unlabeled symbols in the instance and labeled symbols in the model expression. The correspondence determines the label of the symbols in the instance expression.

5) Labeling editor (LE): this interface allows `administrators` to interactively evaluate and correct symbol labeling. Model expressions and the list of instances for the selected model are shown on the left side of the interface, and the matching between the selected model-instance pair is displayed on the main panel (see Figure 5). The computed correspondence between symbols in a model and instance expressions are displayed graphically by line segments linking them. To correct a matching, an extremity of the line segment can be interactively placed over the correct matching symbol.

For expressions with a large number of symbols, line segments may appear cluttered, making visual verification difficult. To facilitate visual inspection in such cases, it is possible to display groups of non intersecting line segments (see Figure 6).

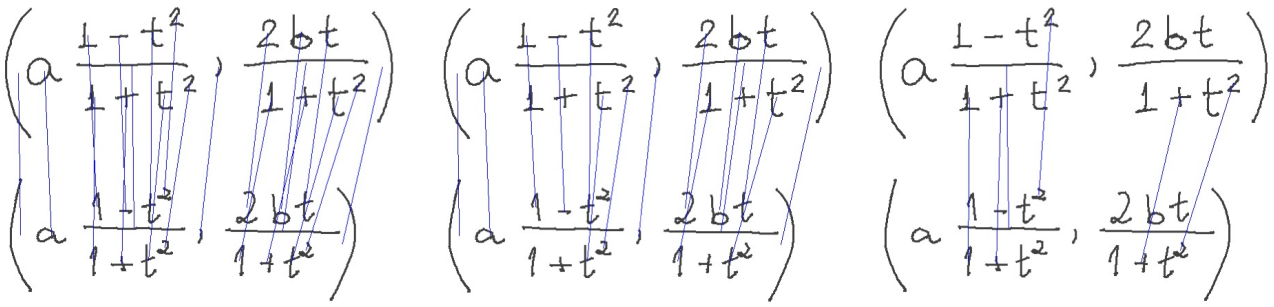
6) Importer/exporter (IED): this component allows model and instance expressions to be imported/exported. Thus, expressions collected in different machines can be joined into a single central dataset. In addition, the whole dataset or a subset of it can be exported as XML format files. Subsets with expressions belonging to specific categories, or written by specific users, or having a specific number of symbols or classes of symbols can be selected for exportation. Datasets with isolated symbol samples can also be exported.

IV. Evaluation

A large dataset consisting of fifty six model expressions has been created in order to evaluate the functionalities and usability of *ExpressMatch*. A total of 25 writers, with background in Engineering or Computer Science, volunteered to transcribe the expressions. Since writing may be a time consuming task, users were asked to write as many instances as their time constraint allowed. For inputting the expressions, an HP tablet PC was used. The average time spent by a user to enter all 56 expressions was about one hour. Through all the collecting process, the threshold used in TBS (for segmentation of symbols) was set to 500 milliseconds, a value that was determined experimentally.

A total of 926 instances were collected. From these, 16 (or 1.7%) were discarded due to segmentation or semantic error. After automatic matching between each of the 910 instances and respective models, they have been visually verified, and incorrect symbol assignments have been manually corrected. A total of 600 incorrect assignments were found (out of a total of 20,010 possible ones), which corresponds to 3% of the symbols. The verification and label correction process took about 5 hours.

In addition, writers were asked to give feedback on how much writing is affected by using the rule described in Section III-A1. Overall, the evaluation is that users had rapidly adapted themselves to the writing rules, not being considered a severe restriction.



(a) (b) (c)

Figure 6. Example of matching with many line segments: (a) with all segments, (b) and (c) two groups of non intersecting segments.

V. Concluding remarks

We have presented *ExpressMatch*, a system that supports the creation and management of ground-truthed online ME datasets. The main features of the system are: (1) it uses a method to segment symbols at writing time, which has been evaluated as not imposing any strong restriction on user's writing; (2) it has user control mechanisms that keep information on which expressions have already been transcribed by each user; (3) it allows incremental input of data (both model expressions, sample instances, and ground-truth data); (4) it allows data collected in different places to be gathered into a single dataset; (5) it automatically labels symbols in transcribed expressions; and (6) it allows visual verification and interactive correction of symbol labeling.

Creation of a dataset with 56 models and 910 sample instances, written by 25 users, have confirmed that *ExpressMatch* provides an easy and efficient way to create ME datasets. Considering all features listed above, we believe *ExpressMatch* has the potential to be used for creating large and expressive datasets in a collaborative and incremental way.

As future research and development, we plan an improvement of the matching approach described in [9], integration of *ExpressMatch* to an online database system, and enhancement of interactive aspects of *ExpressMatch*. *ExpressMatch* as well as the dataset described in this work will be made publicly available at [12].

Acknowledgment

This work is supported by FAPESP (Grant number 2010/04491-0), and by CNPq (Grant number 560165/2010-2), Brazil. The authors thank all writers that have volunteered to input instance expressions.

References

[1] E. Valveny, P. Dosch, A. Winstanley, Y. Zhou, S. Yang, L. Yan, L. Wenyin, D. Elliman, M. Delalandre, E. Trupin, S. Adam, and J.-M. Ogier, "A general framework for the evaluation of symbol recognition methods," *Int. J. Doc. Anal. Recognit.*, vol. 9, pp. 59–74, 2007.

[2] A. Lapointe, "Issues in performance evaluation of mathematical notation recognition systems," Master's thesis, Queen's Univ., 2008.

[3] 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.*, vol. 14, pp. 65–74, 2011.

[4] T. H. Rhee and J. H. Kim, "Efficient search strategy in structural analysis for handwritten mathematical expression recognition," *Pattern Recogn.*, vol. 42, pp. 3192–3201, 2009.

[5] A.-M. Awal, H. Mouchere, and C. Viard-Gaudin, "Towards handwritten mathematical expression recognition," in *Proceedings of the 10th International Conference on Document Analysis and Recognition*, 2009, pp. 1046–1050.

[6] B.-Q. Vuong, Y. He, and S. C. Hui, "Towards a web-based progressive handwriting recognition environment for mathematical problem solving," *Expert Systems with Applications*, vol. 37, no. 1, pp. 886–893, 2010.

[7] N. E. Matsakis, "Recognition of handwritten mathematical expressions," Master's thesis, Massachusetts Institute of Technology, Cambridge, 1999.

[8] L. Wenyin and D. Dori, "Performance evaluation of graphics recognition algorithms: Principles and applications," *International Conference on Pattern Recognition*, vol. 2, pp. 1180–1182, 1998.

[9] N. S. T. Hirata and W. Y. Honda, "Automatic labeling of handwritten mathematical symbols via expression matching," in *Proceedings of the 8th International Conference on Graph-based Representations in Pattern Recognition*. Springer-Verlag, 2011, pp. 295–304.

[10] A. Lapointe and 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.

[11] A.-M. Awal, H. Mouchre, and C. Viard-Gaudin, "The problem of handwritten mathematical expression recognition evaluation." in *ICFHR*, 2010, pp. 646–651.

[12] (2011) The ExpressMatch website. [Online]. Available: http://www.vision.ime.usp.br/demos