Issues in Performance Evaluation: A Case Study of Math Recognition

Adrien Lapointe and Dorothea Blostein
School of Computing, Queen's University, Kingston, Ontario, Canada
adlapointe@gmail.com blostein@cs.queensu.ca

Abstract

Performance evaluation of document recognition systems is a difficult and practically important problem. Issues arise in defining requirements, in characterizing the system's range of inputs and outputs, in interpreting published performance evaluation results, in reproducing performance evaluation experiments, in choosing training and test data, and in selecting performance metrics. We discuss these issues in the context of evaluating systems for recognition of mathematical expressions. Excellent progress has been made in the theory and practice of performance evaluation, but many open problems remain.

1. Introduction

Diverse issues arise in performance evaluation of document recognition systems. These issues are discussed in scattered fashion in the literature. Some issues have been heavily researched while others have received less attention. Here we provide an overview of issues, to stimulate research in the area. Our discussion focuses on the evaluation of systems for recognition of mathematical notation, but similar issues arise in other branches of document recognition.

Performance evaluation can be carried out by an end-user or by a developer. Even though developers and end-users have different goals, many of the issues they encounter are the same. In the math-recognition domain, an end-user carries out evaluation to aid in selecting software that provides a convenient, efficient and accurate means to enter math expressions into the computer. In contrast, a developer carries out evaluation as part of the software design process, to evaluate the performance of the system he or she is developing, and to compare this to the performance of existing systems. We discuss issues that end-users and developers encounter during performance evaluation, and mention existing means of addressing some of these issues. Space does not permit a comprehensive survey of the performance evaluation literature.

2. Defining Requirements

Performance evaluation determines how well a system performs relative to some requirements. A given system will meet some users' requirements very well and other users' requirements less well.

> **Issue 1** System requirements are usually described informally, leaving uncertainty about the domain of math expressions that must be handled, as well as uncertainty about the required accuracy.

An example of an informal requirements description for a math recognition system is "a system for small-scale recognition of algebraic expressions, with a good user interface for correcting recognition results". This description does not define *algebraic expression* or *good user interface*. Is the input handwritten or typeset or both? What range of handwriting variability or font variability must be handled? These topics are discussed further in Section 3, 5, and 6.

3. Characterizing a System's Range of Inputs and Outputs

A fundamental characteristic of a system is its I/O behavior: what domain of inputs is accepted by the system, and what range of outputs is produced? Presently, domains and ranges are characterized informally, often by example. As illustrated by Issues 2-4, research into more precise methods of characterization is needed.

> **Issue 2** Various levels of representation are used for math notation. There are no formal descriptions of these levels, and a variety of formats exist for representing data at each level.

Many document representation formats are in use [8]. Bitmap and PDF files encode mathematical expressions as arrays of pixels or as vectors. Formats such as LaTeX and Presentation MathML explicitly identify symbols and structures. Formats such as

978-0-7695-3725-2/09 $25.00 © 2009 IEEE
DOI 10.1109/ICDAR.2009.247

Maple, Maxima, Mathematica, and Content MathML explicitly encode other aspects related to the meaning of the expressions – for example, whether a particular symbol is a function name or a variable.

Math recognition systems vary in the levels of representation that they accept as input and produce as output. For instance, Infty Reader accepts a document image as input (PDF, JPG, BMP) and produces output in formats including LaTeX, MathML, and XML [1]. In contrast, MathBrush accepts a vector image as input and produces output in Maple [2]. Comparison of these two systems must take these I/O differences into account. To enable comparison, the test data must include document samples represented in a variety of levels and formats.

Issue 3 Representation formats do not enforce uniqueness: given information can be encoded in a variety of ways. Performance evaluation must allow for these equivalences.

There are several ways to represent a mathematical expression within a given format. For example, x_0^2 can be represented in LaTeX as `x^2_0`, `x_0^2`, `x^{2}_{0}`, or `x_{0}^{2}`. Analogous equivalences occur in other formats such as MathML [5]. When system output is compared to ground truth, these equivalent representations must be treated as equal. This problem can be addressed by defining a *canonical form*, a standard that specifies a unique way to represent a given expression within a representation format [4][5]. Some equivalences require extensive domain knowledge, for example the algebraic equivalence of $a+b+c$ and $c+a+b$.

Issue 4 Performance evaluation should include a characterization of the system's *coverage* (the range of documents that the system can accept as input). However, it is difficult to determine the coverage of a document recognition system.

A math recognition system is designed to handle some subset of math notation. It is misleading to measure the performance of a system by testing it on symbols, structures or dialects that the system was not designed to recognize. Ideally, the coverage should be described as part of the performance characterization. Some systems have high performance on a small domain while others have lower performance on a larger domain. Which of these is preferable depends on the user's requirements.

The nature of the input is an important aspect of coverage, and must be considered when interpreting performance measures. How much noise, and what kinds of noise, does the input contain? Is the input handwritten or typeset? Handwritten notation is less regular, with more overlap among symbols and greater variability in symbol placement. Thus, handwritten input gives rise to more errors in segmentation, symbol recognition, and structure analysis.

Coverage is difficult to quantify. Currently, we do not have techniques to precisely define the entire space of math notation, or to precisely delineate subspaces of math notation. Also, it is difficult to determine the coverage of a math recognition system. We have experimented with four sources of information that can be used to estimate coverage [18].

- Published papers Coverage can be estimated from the literature, although published descriptions are often incomplete. This is illustrated in Table 1.
- Execution of the system Coverage can be estimated by observing system response to test data that contains a variety of symbols and structures.
- Source code Coverage can be estimated by studying the source code. It may be difficult to locate information about coverage within the code.
- Training options Coverage can be estimated from the system's training options. For example, to train recognition of handwritten symbols, the user may be asked to write samples of specified symbols. These symbols are part of the system's coverage.

These methods are helpful, but obtaining complete information about coverage remains difficult.

TABLE 1. COVERAGE OF MATH RECOGNITION SYSTEMS, AS REPORTED IN PUBLICATIONS.

	Symbols	Structures	Functions
Taki-guchi *et al.* [23]	Latin alphabet Greek alphabet Digits + − × ÷ /	Horizontal, fraction, subscript, superscript, over, under, integral, sum	acos arccos arcsin and 33 others
Kos-mala *et al.* [17]	Latin alphabet Digits + − · : / — ^ √ ∑ ∏ ∫ , ´ = < > ≤ ≥ → ↔ ≈ ! ∞ α β λ μ Δ π ω ε τ ϕ ( ) [ ] { }	Not specified	Not specified
Ash-ida *et al.* [6]	Latin and Greek letters, digits, double $\mathbb{A}\mathbb{J}\mathbb{P}\mathbb{Z}$ script $\mathcal{A}\mathcal{E}\mathcal{F}\mathcal{Y}$ fraktur $\mathfrak{A}\mathfrak{B}\mathfrak{F}\mathfrak{G}$ ( ) → ∪ etc.	Horizontal, fraction, left and right sub- and superscripts, matrix, square root	lim sin
Oka-moto *et al.* [21]	All LaTeX symbols except user defined ones.	Horizontal, fraction, left and right sub- and superscripts, matrix, square root	Not specified

4. Interpreting Published Performance

Published performance measures are a valuable source of information. Issues 5 and 6 arise in interpreting this information.

Issue 5 Reported performance results are often based on limited test data.

Performance evaluation is time-consuming, and authors are understandably limited in the size of data sets they use for testing. Performance measures reported for one data set may not be representative of performance on other data sets; this problem is exacerbated when test sets are small.

Issue 6 It is difficult to compare the performance measures reported for different systems.

Comparison of performance measures is difficult because of the variety of metrics and data sets that are used to measure performance [28]. This is illustrated in Table 2. Some metrics, such as *symbol recognition rate* and *expression recognition rate*, are used by several authors, allowing comparison of those aspects of system performance.

TABLE 2. A SAMPLE OF METRICS USED TO REPORT PERFORMANCE OF MATH RECOGNITION SYSTEMS (CONDENSED FROM [18]).

Authors	Metrics
Ashida *et al*. [6]	– Symbol recognition rate
Chan and Yeung [10]	– Symbol recognition rate – Expression recognition rate – Operator recognition rate – Integrated performance measure
Garain and Chaudhuri [12]	– Global performance index – Average performance index
Kosmala *et al.* [17]	– Computing time
Okamoto *et al.* [21]	– Expression recognition rate – Character recognition rate – Structure recognition rate
Takiguchi *et al.* [23]	– Character recognition rate
Zanibbi *et al.* [28]	– Baseline recognition rate – Token placement rate – Expression recognition rate

5. Reproducing Performance Evaluation Experiments

Reproducibility of experiments is a cornerstone of the scientific method.

Issue 7 Performance evaluation techniques that are used in the literature are not always reported in enough detail to allow reproduction of the experiment.

The following paragraphs discuss a few reasons why performance evaluation experiments may be difficult to reproduce. We illustrate with examples drawn from the literature, but do not intend to single out any particular authors. Many publications (including our own) report insufficient detail about performance evaluation techniques. Research is required to develop methods for more precisely and fully defining experimental conditions.

Sometimes reproducibility is hampered by insufficient detail in the definition of metrics. For example, a performance evaluation metric based on *levels* is used in [12]. As defined by the authors, *levels* are horizontal lines that are superimposed on an image of a mathematical expression: the level lines are located so that they intersect symbol centroids, with nearly-collinear centroids grouped on the same level line. The number of level lines provides a measure of the complexity of a mathematical expression. However, reproduction of the reported experiments is impossible without a more precise definition of *nearly collinear*.

Sometimes reproducibility is hampered by insufficient detail about the definition of *correct* recognition. For example, in reporting symbol recognition performance, authors sometimes forgive confusion among highly-similar symbols. When this is done, the results are not reproducible unless a complete set of forgiven symbols is provided. For lack of space, authors sometimes only report representative examples of confusions that are forgiven, e.g. [6].

A precise definition of the document recognition task is required for reproducing and comparing performance measures. The task definition must specify the training and test data, as well as the performance metrics. Document recognition contests have provided excellent progress in this area. The contests provide standardized task definitions for tasks such as dashed-line detection, raster to vector conversion, arc segmentation, symbol recognition, page segmentation, handwriting segmentation, and Arabic handwriting recognition [11]. Performance protocols used in these contests are described in publications such as [26]. This work provides important insights into the difficult and multi-faceted problem of task definition.

Issue 8 Third-party performance evaluation depends on the availability of source code and executable code.

In *black box performance evaluation*, system output is compared to ground truth, thus evaluating the product, but not the process [13][24]. Sometimes, blackbox evaluation is applied to individual system components; for example, symbol recognition may be evaluated separately from structure analysis [6][10][12]. Executable code must be available for black box evaluation to be carried out. Most performance measures reported in the literature are the result of black box evaluation.

In contrast, *white box evaluation* examines the internal workings of the system. For example, a history of recognition hypotheses may be recorded [29]. White-box evaluation requires access to source code. If the original source code is not available, the

system can be reimplemented based on published descriptions.

Issue 9 Parameter settings have a large effect on system performance. It is often difficult to choose appropriate parameter settings, or to reproduce the parameter settings used in reported experiments.

Optimization of system parameters is an important factor in quantifying performance [16]. A system with few parameters is preferable, because it requires less tuning, and is likely to have reasonably good performance over a wide range of inputs. In any case, the number of parameters and the method of choosing parameter values must be clearly reported.

6. Choosing Training and Test Data

The choice of training data and test data is critical to the outcome of performance evaluation. Data sets should be large and representative, including ground-truth information. Publicly-available data sets are essential for supporting performance comparisons.

Issue 10 There is a shortage of large, representative, publicly available, ground-truthed data sets.

Standardized data sets offer many advantages, allowing experiments to be reproduced, and allowing results to be directly compared [12][27]. The alternative is to create custom data sets, which inadvertently might be tuned to a particular algorithm [16]. Examples of publicly available data sets for math notation include UW-III [20] and Infty [1].

Ground truth is an essential component of a data set Several ground truth representations are needed for a given input, to reflect the different data-representation levels and formats (see issue 2). An estimate of the accuracy of ground truth is needed in order to interpret performance measures. Some data sets contain highly accurate ground truth, created by combining multiple, independently-produced representations of ground truth [22][25]. Different people may have different concepts of the ground truth for a given document image [14].

The creation of a ground-truthed data set is time-consuming and expensive [25][27]. For example, Baird and Casey report that the three CD-ROM University of Washington database cost about two million dollars [7][20]. Copyright considerations can be a barrier to making a data set available e.g. [6].

Large quantities of document data are available in internet paper repositories. Some repositories provide document images without ground truth. For example, the digital mathematics library provides scanned pages of math notation [15]. In other repositories, document images are generated from ground truth files. For example, the preprint repository ArXiv provides LaTeX source files for over half a million papers from different fields of science; since the PDF files are generated from a LaTeX source, the document images do not include noise and variability that is found in scanned documents [3]. Noise and distortion can be introduced algorithmically, through application of document noise models [7][25]. This method – generating document images from ground truth, and adding synthetic noise – provides ground truth cheaply, and therefore allows relatively inexpensive creation of large document databases.

Issue 11 A document data set should be representative of some domain of document images. However, domains are difficult to describe, and the representativeness of a data set is difficult to assess.

Representative data sets are used in testing, because it is impossible to test performance on every document that the system is supposed to accept as input [13]. A major difficulty is that we lack techniques for defining domains of documents, or degrees of coverage of these domains. For example, in mathematical notation, a dataset may focus on a subset of mathematical notation, such as arithmetic expressions or calculus expressions. Currently, it is not possible to describe the degree of coverage of such a data set; instead, users look at the contents of the data set to form their own informal impression of the coverage. Future research may provide improved ways of characterizing the coverage of a data set. This could include information on the frequency of certain operators or structures, as well as the prevalence of particular types of noise.

The best method of creating a representative data set is to ensure diversity when collecting the data. Suggestions for achieving this include making the data sets large [19][25]; covering a variety of acquisition parameters, deformations, noise and degradations [25]; and varying the frequency of equations per page, the nature of expressions (geometric structures and layouts), the typesetting methods, the page layout, and the document age [12].

7. Selecting Performance Metrics

After matching system output with ground truth, performance is quantified using metrics.

Issue 12 It is difficult to assess the quality of performance metrics and to select the best metrics for quantifying system performance.

Various metrics are in use (see Table 2). Criteria for choosing between metrics include the following. Metrics should measure the extent of fulfilment of the system's objectives [27]. They should be comprehensive, encompassing all the system objectives [27]. Metrics should be compatible with human evaluation [27], and should be known and accepted by the community [25][27].

Issue 13 Each metric reflects certain performance aspects. It is not clear how best to combine metrics.

Many aspects of system performance have to be evaluated, including correctness, efficiency, usability, comprehensiveness of coverage, quality of coverage, and robustness to noise. As a result, a variety of metrics are used [9][25]. Several metrics can be reported in parallel, or metrics can be combined into a conglomerate metric. Proper combination of metrics is difficult [27]. Commonly, metrics are combined using weighted sums. Unfortunately, the choice of weights is subjective and may vary from one author to the other, detracting from the objectiveness and portability of the performance evaluation results.

8. Conclusion

Evaluating the performance of document recognition systems involves defining requirements, characterizing the system's range of inputs and outputs, interpreting published performance evaluation results, reproducing performance evaluation experiments, choosing training and test data, and selecting performance metrics. We have studied these issues in the context of evaluating systems for recognition of mathematical expressions; similar issues arise in other branches of document image analysis. Research progress is notable in areas such as the definition of document noise models, the creation of publicly available document data sets, and the precise specification of document analysis tasks (for document recognition contests). An important open problem is to develop terminology and notation for describing and delineating a set of documents. Such terminology is needed to (1) describe the domain and range of a document recognition system, and (2) describe the contents of a document data set. Other open problems include precisely defining system requirements, improving the reproducibility of performance evaluation experiments, and choosing and combining performance metrics.

Acknowledgments

Financial support from the Natural Sciences and Engineering Research Council of Canada and from the Xerox Foundation is gratefully acknowledged.

References

[1] Infty. www.inftyproject.org/en/index.html, April 23, 2008.

[2] MathBrush. www.cs.uwaterloo.ca/scg/mathbrush, Apr 23, 2008

[3] arXiv. http://arxiv.org/, November 1, 2008.

[4] M. Altamimi, A. Youssef, "A More Canocial Form of Content MathML to Facilitate Math Search," *Proc. Extreme Markup Languages,* 2007.

[5] D. Archambault, V. Moço, "Canonical MathML to Simplify Conversion of MathML to Braille Mathematical Notations," LNCS Vol. 4061, pp. 1191–1198, Springer, 2006.

[6] K. Ashida, M. Okamoto, H. Imai, T. Nakatsuka, "Perf. Eval. of a Math. Formula Recognition System with a Large Scale of Printed Formula Images," *Proc. DIAL'06*, 321–331, Apr. 2006.

[7] H. Baird, M. Casey, "Towards Versatile Document Analysis Systems," LNCS Vol. 3872, pp. 280–290, Springer, 2006.

[8] D. Blostein, R. Zanibbi, G. Nagy, R. Harrap, "Document Representations," *Proc. GREC'03*, pp. 3–12, July 2003.

[9] P. Calingaert, "System Performance Evaluation: Survey and Appraisal," CACM, **10**(1):12–18, January 1967.

[10] K-F Chan, D-Y Yeung, "Error Detection, Error Correction and Performance Evaluation in On-Line Math. Expression Recognition," *Pattern Recognition*, **34**(8):1671–1684, Aug. 2001.

[11] Document Recognition Contests: www.icdar2007.org/competition.html

[12] U. Garain, B. Chaudhuri, "A Corpus for OCR Research on Mathematical Expressions," *IJDAR*, **7**(4):241–259, Sept. 2005.

[13] G. Guida, G. Mauri, "Evaluation of Natural Language Processing Systems: Issues and Approaches," *Proceedings of the IEEE*, **74**:1026–1035, July 1986.

[14] J. Hu, R. Kashi, D. Lopresti, G. Nagy, G. Wilfong, "Why Table Ground-Truthing is Hard," *Proc. ICDAR 2001*, Seattle, Washington, Sept. 2001, pp. 129-133.

[15] A. Jackson, "The Digital Mathematics Library," *Notices of the AMS*, **50**(8):918–923, September 2003.

[16] E. Keogh, S. Lonardi, C. Ratanamahatana, "Towards Parameter-Free Data Mining," *Proc. KDD 2004*, 206–215, 2004.

[17] A. Kosmala, S. Lavirotte, L. Pottier, G. Rigoll, "On-Line Handwritten Formula Rec. using HMMs and Context Dependent Graph Grammars," *Proc. ICDAR'99*, pp. 107–110, 1999.

[18] A. Lapointe, "Issues in Performance Evaluation of Math. Notation Recognition Systems," *MSc thesis, Queen's Univ.,* 2008.

[19] S. Mao, T. Kanungo, "Empirical Performance Evaluation Methodology and its Application to Page Segmentation Algorithms," *IEEE PAMI*, **23**(3):242–256, 2001.

[20] University of Washington UW-III English/Technical Document Image Database. CD-ROM, 1996.

[21] M. Okamoto, H. Imai, K. Takagi, "Performance Evaluation of a Robust Method for Mathematical Exprression Recognition," *Proc. ICDAR'01*, pp. 121–128, 2001.

[22] I. Phillips, A. Chhabra, "Empirical Performance Evaluation of Graphics Recognition Systems," *IEEE Trans. Pattern Analysis and Machine Intelligence*, **21**(9):849–870, 1999.

[23] Y. Takiguchi, M. Okada, Y. Miyake, "A Fundamental Study of Output Translation from Layout Recognition … for Mathematical Formulae," *ICDAR'05*, **2:**745–749, 2005.

[24] M. Thulke, V. Märgner, A. Dengel, "A General Approach to Quality Evaluation of Document Segmentation Results," LNCS Vol. 1655, pp. 43–57, Springer, 1999.

[25] E. Valveny et al., "A General Framework for the Evaluation of Symbol Recognition Methods," *IJDAR*, **9**(1):59–74, 2007.

[26] L. Wenyin, D. Dori, "A Protocol for Performance Evaluation of Line Detection Algorithms", *Machine Vision and Applications, Special Issue on Performance....,* **9**(5/6): 240-250, 1997.

[27] L. Wenyin, D. Dori, "Performance Evaluation of Graphics Recognition Algorithms: Principles and Applications," *Proc. ICPR'98*, **2**:1180–1182, 1998.

[28] R. Zanibbi, D. Blostein, J. R. Cordy, "Recognizing Mathematical Expressions Using Tree Transformation," *IEEE PAMI*, **24**(11):1455–1467, 2002.

[29] R. Zanibbi, D. Blostein, J. R. Cordy, "Historical Recall and Precision: Summarizing Generated Hypotheses", *Proc. ICDAR'05,* **1**:202–206, 2005.