When is a Problem Solved?

Daniel Lopresti
CSE Department
Lehigh University
Bethlehem, PA 18015, USA
Email: lopresti@cse.lehigh.edu

George Nagy
ECSE Department
Rensselaer Polytechnic Institute
Troy, NY 12180, USA
Email: nagy@ecse.rpi.edu

***Abstract*—Open problems are defined differently in document image analysis than in the physical sciences, theoretical computer science, or mathematics. Instead of a formal definition, problems in DIA are stated in terms of automation of an application area (*e.g.*, postal address reading) or a scientific subfield (*e.g.*, image compression). The notion of a successful solution may be based on (1) the relative accuracy of automated vs. expert solutions (given specific data and degree of manual tuning); (2) the distinguishability of automated output from human output (a Turing Test); (3) the degree of current community interest (via conferences and journals); and/or (4) economic considerations. Because of the lack of formal definition for DIA problems, heuristics predominate over provably correct algorithms, and full disclosure of implementation details as well as populations and samples is essential. Results on available test sets are often only tangentially related to motivating applications. In addition, interest in automating certain tasks has been evolving rapidly as a result of advances in technology. Further community discussion of these issues may accelerate progress and symbiosis with allied disciplines.**

***Keywords*-document analysis; pattern recognition; performance evaluation;**

I. Introduction

As scientists working in document analysis, we dedicate our careers to the goal of developing methods for solving pressing problems in our field. It seems natural, then, to ask ourselves "When is a problem solved?" There is a presumption that the problems we study must be solvable; if they were fundamentally intractable, then we would be wasting our time. We can point to the fact that humans can (often) easily perform the kinds of tasks we try to get machines to master. We must likewise believe that it would be a waste of time to continue working on a problem after an effective solution has already been found. There are enough challenging problems that remain open, with new ones arising regularly, that there should be no need to devote energy to problems that have already been addressed in a satisfactory fashion. When asked, we suspect that each researcher would have his/her own opinion, but we are not aware of a community-wide consensus that all would agree on. Hence, the question seems like an important one to consider. When, then, is a problem solved?

Is this a goal that can be phrased in a rigorous, mathematical way? Or is it more of an ill-defined notion that is answerable only via social convention? We might begin by pondering the oft-stated claim that OCR is a "solved" problem. This is an assertion we hear mainly from those outside our community. In response, we may quickly point out that, yes, there exist software products that will obtain near-perfect accuracy on clean, well-formatted office documents written in one of several languages, but that major challenges still exist for noisy inputs, handwriting, complex layouts, and commercially less-lucrative languages. It is conceivable that with sheer engineering effort and no new scientific breakthroughs, OCR for some other languages (but not all) could be brought to the same level of performance. So, if OCR is not solved now, what would it take to solve it?

Can we even agree on what a "problem" is in document analysis? It is certainly not a theorem which can be proved or disproved like the Four Color Problem, or verified by experiment like the Second Law of Thermodynamics. Nor is it the specification of an algorithm, such as is commonly formulated for sorting integers, enumerating the connected components in a bitonal image, or finding the collinearities among a set of (x, y, z) coordinates. For this discussion, it is expedient to consider a problem simply as the (partial) automation of a task originally performed by reading, writing, and perhaps typing. Whether this problem is solved must then be related in some way to the degree of success achieved in automating the task – and therein lies the rub.

When reviewing the literature in our field, it is common to see phrases like "The results we report seem promising." On the other hand, we never encounter claims that a problem has been solved (and, if we did, we suspect it would immediately raise suspicion). Assuming this is not merely a sign of excess modesty, is the burden of proof set too high? Or is the concept of solving a problem not well understood?

By asking such questions, we can hope to achieve a better picture of what it is we are trying to accomplish as a field, as well as what we should be trying to communicate to each other and to our peers working outside of document analysis. Our goal for this paper is to initiate discussion that will hopefully lead to greater awareness and, ultimately, consensus on what ought to be a key concern in our research.

II. Defining the Question

It is a commonly held belief that, in the end, everything we do can be couched in mathematical terms. There is tremendous value in mathematical rigor, and we shall discuss such considerations in the next section. First, though, we attempt to enumerate what we believe are the different common viewpoints one might hear when asking members of our community "When is a problem solved?"

A. The "endless pursuit of perfection" viewpoint

One possible (intuitive) view is the following: "A problem is solved if there is a method which has been widely publicized and documented and freely available to the community which achieves 100% accuracy on any input it receives."

This is a wonderfully general and yet highly problematic definition, as we shall show, and yet it is arguably the definition that some researchers may have in mind when they are reluctant to declare a problem solved.

Let us break down the analysis of this definition to see what might be good and bad about it:

1) "... widely publicized and documented and freely available to the community ..." This seems reasonable. If there is a proprietary solution, or one which is known internally to some organization but to no one else, then the problem is not solved.
2) "... which achieves 100% accuracy ..." There are two problems with this part of the definition. The first is what we mean by "accuracy." Should errors made by the original author be preserved? The second is whether the goal of 100% accuracy must be achieved. We might try to define accuracy as agreement with a human expert. However, even experts disagree. So the question then becomes, agreement with which human expert? And why should we need to be 100% in agreement, if different human experts disagree?
3) "... on any input it receives." This is another problematic statement. Input that is sufficiently degraded will always result in "GIGO" (garbage-in, garbage-out).

Perhaps the problems we work on are so intrinsically hard that it will always be possible to improve upon past techniques, asymptotically approaching, but never quite achieving, perfection. This may make us feel good – our jobs are assured – but it does not seem like a healthy attitude for scientists attempting to advance a field.

B. The "Turing Test" viewpoint

One problem with the proceeding definition is that it assumes there is a single, well-defined "ground truth." For all but the most trivial problems, we can expect human experts to disagree. We could, then, collect opinions from multiple experts and allow the algorithm the benefit of the doubt by selecting the expert version closest to its output for each input that it processed. This approach, however, leads to lack of consistency across different inputs.

We can turn the question around by asking the question somewhat differently: is the output from the computer indistinguishable from the output from a human expert? We are all pretty good at recognizing the kinds of errors that machines make and hence it seems possible that we could look at a set of outputs and decide whether it came from a human or a machine. This is, of course, the definition of the classic Turing Test, extended to where the domain of interaction is the subfield of machine vision and natural language understanding that constitutes document analysis. We have not heard anyone suggest this idea as a way of judging the performance of algorithms in our field, but it is obvious adaptation of the idea that machines should try to emulate a human expert.

According to this viewpoint, then: "A problem is solved if there is a method which has been widely publicized and documented and freely available to the community which generates output for a given input that a human judge cannot reliably distinguish from the output of a human expert."

This is an elegant definition. What prevents it from being useful in practice is the labor involved in making the determination, since the judge in a Turing Test must be a human. (We could, however, consider whether it might be possible to build a classifier that would distinguish machine-made error patterns from human error patterns – see [3] for a similar idea applied to the problem of validating synthetic degradation models.)

C. The "as good as it gets" viewpoint

We might acknowledge both that perfection is impossible for some tasks of interest, and that human performance is unreachable. This does not mean we should not try to tackle such problems, just as theoretical computer scientists do not shy away from provably hard optimization problems, but rather turn toward developing approximation algorithms with guaranteed performance bounds.

The definition in this case would be: "A problem is solved if there is a method which has been widely publicized and documented and freely available to the community which performs better than any other method, and which cannot be further improved without investing excessive resources."

This seems like a good definition although, of course, the end game is not well defined. Perhaps we can declare a problem solved if, after a certain period of time, no one has been able to improve upon a particular algorithm. (It has been previously noted, however, that for any problem, a dataset can be constructed where a new algorithm will outperform all existing techniques [4]. Perhaps we need added verbiage in our definition to rule out such manipulations.)

It may also be that a solution exists which can be manually tuned to any given application, but no universal method is available. Is it important to continue to seek a fully general approach, or can we comfort ourselves in knowing that with some added engineering effort, the problem is solvable?

D. The "good enough to get the job done" viewpoint

If we acknowledge that nearly all of the methods we develop are embedded in bigger systems, often as one stage in a computational pipeline, we may come to realize that the step we are working on is not the determinant in the overall performance of the system.

For example, when we insert a handwritten character recognizer into a postal address reading system, we may find there is no significant difference in mail routing costs between a shape classifier that distinguishes '0' (zero) from 'O' (oh) with 70% accuracy versus one that achieves 80% accuracy; the later-stage contextual analysis will determine the correct outcome in either case.

A classic example is that segmenting printed text into individual characters is harder than recognizing individual characters after they have been correctly separated. Yet the second problem has attracted far more attention.

This viewpoint is rarely studied because of the difficulty in implementing end-to-end systems in academia.

E. The "pragmatic" viewpoint

Perhaps a problem is solved when papers about it start getting regularly rejected from our conferences and journals. Then people will naturally drift away from this line of research into something else because of the desire to work on "hot" problems. This may be one of the main reasons for employing a peer review process in science.

There is a danger here, however, because it seems possible to get a paper published on just about any topic by employing a number of "tricks" that we learn over time. The proliferation of conferences and journals – some of which are run as money-making ventures – aggravates the danger. Document processing, nevertheless, has far fewer venues than computer vision, artificial intelligence, or multimedia. It has been, and remains, a relatively small field of study.

Pragmatic considerations can also be used to draw the fine line between applied science and engineering. There is no doubt that a great deal of engineering effort must be applied after the basic scientific techniques have been developed to produce a working solution. This is beyond what most research groups can do. For example, the number of written languages in the world is estimated to be in the thousands. OCR may be solved for some languages, but we can rightly claim it is not yet solved for other languages, even though the same techniques may apply, until the necessary large-scale engineering effort has been devoted to those languages. Is a problem solved if the techniques to solve it most likely exist, but no one has taken the steps to put them into practice for the task in question?

III. Scientific Considerations

The previous discussion suggests there is a diversity of viewpoints concerning when we might decide that a problem has been solved. Here we consider some of the scientific considerations that come into play, both with regard to the way problems are defined as well as to the scientific methodology that is applied when trying to solve them.

A. Problem definition

Where does our problem "sit"? The way in which we refer to it in our papers creates an awareness and expectations from those reading the paper. This directly relates to beliefs regarding what is necessary to solve the problem, and what level of performance is required.

Based on common conventions practiced in the field, it may make sense to categorize a problem based on application areas: postal address reading; bank check reading; mail room functions; book OCR (Google Books, Million Book Project); medical forms / records processing; archival engineering drawing conversion; paper-based election technologies (op-scan ballots); historical documents; forensic document analysis.

Or by "scientific" subfields: image capture (scanners, cameras); image processing; classification, pattern recognition, machine learning; data mining; natural language understanding; computer graphics and visualization; and human-computer interaction.

B. Methodologies

When conducting experiments, we want to be as rigorous as possible. Our goal is to convince other experts that we have solved a particular problem, or at least that we have made substantial progress. Hence, good methodologies are key. These clearly interact with our discussion regarding the basic nature of the question, "When is a problem solved?"

1) Populations and samples: Performance figures like error, reject, or retrieval rates are of interest only with regard to populations rather than particular samples. Many statistical felonies and misdemeanors are related to violations of the fundamental principle of estimating population statistics from a random sample. Because throw-away sampling of an entire population is expensive, most DIA experiments are conducted on convenience samples designed to compare methods without representing any clearly defined population.

Examples of populations include:

1) All pages in all Google books.
2) All ballots cast in the 2008 U.S. Presidential Election in the state of Minnesota.
3) All tables at 10 specific national statistics websites.
4) Every formula in a new, complete, high quality scan of Abramowitz and Stegun's Handbook of Mathematical Functions with Formulas, Graphs, and Mathematical Tables (National Bureau of Standards edition, Tenth Printing 1972, with corrections).
5) Every graph in above.
6) Every page of every issue of PAMI from 2001-2010.
7) Every page of every article in PAMI from 2001-2010.

8) Every schematic logic diagram in Wakerly Digital Design 4th edition (2006) converted to bitmap from PDF files of illustrations from the publisher's website.
9) Digital photos of every envelope received by some academic department or business in 2008.
10) Scans of every form handled by all secretaries in an academic department in 2010.

Good experimental procedure would dictate selecting a random sample from any of the above populations. Well-defined stratified sampling is also admissible. The existence of standard test sets, on the other hand, does not guarantee progress if they are overused to the point that their entire contents are known implicitly by those who are developing the methods to process them. For a more complete discussion of document census, see [5].

Approaches for applying Web 2.0 technology for the creation and maintenance of datasets and for automating the unbiased testing of document analysis algorithms hold promise for addressing many of these issues [2].

2) Algorithms, heuristics, and implementations: Most document processing systems are built on heuristics rather than algorithms, although the latter term is applied to both. In contrast to an algorithm, a heuristic (*cf.* soft computing) is not guaranteed to work on any particular set of inputs. Because of the seemingly insurmountable difficulty of defining analytically real (or even realistic) inputs, few algorithms have been developed specifically for document processing.

Algorithms developed in other contexts, such as classification algorithms, may be provably correct, but a given implementation may be flawed. Is a problem solved when we have code that everyone can run? Or is it when we think we have a method that should do the job in theory, but no one has been able to create a bug-free implementation? Some algorithms are mathematically optimal, but devilishly hard to program. And any program of more than 10 lines is almost certain to contain bugs. In theoretical computer science, we say that a problem is solved if we know an efficient algorithm, but this definition can be used only for some of the components in a document processing pipeline.

Well-defined algorithms exist for certain techniques, *e.g.*, mathematical morphology, connected component analysis (with various measures of connectivity), wavelets and other transformations (Haar, Fourier, Rademacher), but these are not in themselves DIA tasks. Binarization, on the other hand, is an example of an ill-defined problem.

Classification in vector spaces has clearly defined criteria given fixed training and test sets in terms of error, correct, and reject rates (*cf.* receiver operating curves), but without explicit description of the built-in assumption, it is impossible to predict results for different training and test sets.

Clustering and unsupervised learning can use many criteria for grouping. Methods that produce all groupings, from all-in-one to all-separate, are easier to compare, but are useless in most applications.

There seems to be a dichotomy between what can be formulated in a neat mathematical framework, and what is needed to solve document analysis problems.

3) Desirable criteria for solutions: In this section, we enumerate some specific criteria that are easily tested for but often ignored when developing algorithms that purport to solve problems in our field.

- Invariance to 90-degree rotations (often claimed, but seldom demonstrated).
- Invariance to resolution reduction within given limits. (Multiresolution techniques exploit lack of invariance.)
- Invariance to remapping (either 1:1 or m:1) gray or RGB values. Desirable because the original mapping from reflectance to gray values is a property of transducer settings which are never quoted in papers.
- Invariance to a limited range of threshold settings for globally binarized gray scale images.

Lossy pre-processing can be defined as global application of document transformations without explicit differentiation of types of regions. Is it necessary or desirable? Document parametrization may be a superior alternative.

C. Problems that may be solved

In this section we list some problems that may (or may not be) solved. Our goal is not to create controversy or suggest that further work is unnecessary. Rather, as concrete examples of problems that have been studied extensively and that are widely understood, they can form a basis for discussions about the larger questions we are asking. We note that many of these might be considered pre-processing.

- Binarization. There is no generally-agreed definition of success applicable to multi-source gray-scale or color documents with text and graphics. New algorithms are demonstrated on a small sample and judged visually.
- Document segmentation into two to five component types. Again, there is no universal agreement on what would constitute success. What should be done with background that falls inside of foreground? Should regions be pixel, polygon, or rectangle based?
- Thinning and skeletonization. Several hundred algorithms have been published, but proof exists that not all generally-agreed properties can be simultaneously satisfied. In contrast, Medial Axis Transform and Chamfer Distance are well defined.
- Printed and handwritten paragraph, line, and word finding. Large databases exist for comparing algorithms, but relationship between these databases and any operational application is tenuous. Different conventions exist for handling overlapping script. Most handwritten test data is obtained by having subjects copy templates. Quality control on scanning, especially preservation of calibration targets scanned with the same settings, is usually deficient. *E.g.*, this data is not readily available for Google Books or Million Book Project sources.

- Printed or handwritten skew estimation and calibration. Success rates depend on granularity of estimates: page, column, paragraph, line or word. Success rates are also highly dependent on the source of the test samples.

D. Problems that are becoming less interesting

There are a variety of problems for which research seems to have plateaued, including non-domain-specific OCR on short passages of printed text, document image compression, text compression, thinning and skeletonization, skew estimation, arrow-head recognition, and logo recognition.

Other problems appear to be fading due to economic, business, or societal changes. These include bank check reading, postal address reading, income tax form interpretation, printed map conversion, conversion of circuit diagrams (which, in contrast to plans of tunnels, bridges or transmission lines, have a short life span), compiling concordances, processing X-ray film, and inked signature analysis.

Then there are tasks which, while holding some technical interest, do not present a cost-benefit ratio that justifies automation. *E.g.*, spending 12 months to develop a specialized OCR engine to transcribe a valuable manuscript when it would take only two months to enter the text manually.

It is important, too, to recognize that sometimes some problems do not need to be solved in every context. Determining reading order, for example, appears to be a key step in document layout analysis, but when the ultimate application is page-level vector space information retrieval, having the reading order is unnecessary.

Finally, some problems are better tackled by giants (*e.g.*, Microsoft, Google, government agencies) than by the small research groups that predominate in academia.

IV. Social Considerations

It seems unlikely we will decide a problem is solved based purely on mathematical criteria – it will have to be based on social conventions. Among the features that would be needed for the community to become convinced that a problem is solved, the existence of clear, complete reports on replicable methodology and experiments seem vital.

In a celebrated 1979 article, three respected computer scientists called attention to the essential differences between mathematical proof and formal program verification [1]. They emphasized the importance of the social process, as opposed to a sequence of irrefutable minute steps, in the acceptance of the validity of propositions and theorems. Among their several arguments that bear on the solution of problems in document analysis, one that stands out is the importance of successive joint modifications of programs and specifications, as opposed to the static nature of mathematical and algorithmic problem statements. The technology-driven evolution of specifications in document analysis is reflected by the sea- change in the size, composition and variety of the test data sets used to measure performance improvements, as well as in the growth in complexity and difficulty of our competitions.

As is the case for most engineering disciplines, all of our problems can be solved *to some extent*. We can recognize printed or handwritten words, produce sounds from printed music notation, separate equations and illustrations from narrative text, extract data from web tables, and detect forgeries ... to some extent. Yet given any new collection documents, we cannot predict with any degree of precision or certainty how well we can process it.

V. Discussion

In this paper, we have attempted to raise some important questions arising from the basic desire to know when our research has succeeded. In doing so, we hope to spark productive discussion that will ultimately help the field of document analysis progress.

It is not completely satisfying to offer an assessment of the state of our field and not propose some way of measuring concretely where we stand. We are well aware, however, that any such suggestion is rife with exactly the same issues we have been raising throughout this paper. What criteria should we use to gauge our progress? We posit that there is an answer to this question within the community, and we look forward to the ensuing debate.

We conclude by noting that our sole purpose is not just to solve problems, but also to create new knowledge and to expand basic understanding of natural phenomena. Research on document image analysis has borrowed much from allied fields, and ideally we will find ways to reciprocate in kind.

Acknowledgment

Daniel Lopresti acknowledges support from a DARPA IPTO grant administered by Raytheon BBN Technologies.

References

[1] R. A. DeMillo, R. J. Lipton, and A. J. Perlis, "Social Processes and Proofs of Theorems and Programs," *Communications of the ACM* 22(5): 271-280 (1979).

[2] B. Lamiroy, D. Lopresti, H. Korth, and J. Heflin, "How Carefully Designed Open Resource Sharing Can Help Expand Document Analysis Research," *Proceedings of Document Recognition and Retrieval XVIII*, San Francisco, CA, January 2011, pp. 78740O-1-78740O-14.

[3] Y. Li, D. Lopresti, G. Nagy, and A. Tomkins, "Validation of Image Defect Models for Optical Character Recognition," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 18, no. 2, February 1996, pp. 99-108.

[4] G Nagy, "Candide's Practical Principles of Experimental Pattern Recognition," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 5, no. 2, March 1983, pp. 199-200.

[5] G. Nagy and P. Sarkar, "Document Style Census for OCR," *Proceedings of the First International Workshop on Document Image Analysis for Libraries*, Palo Alto, CA, January 2004, pp. 134-147.