Comparing Approaches to Mathematical Document Analysis from PDF

Josef B. Baker, Alan P. Sexton, and Volker Sorge
School of Computer Science
University of Birmingham
Email: J.Baker|A.P.Sexton|V.Sorge@cs.bham.ac.uk
URL: www.cs.bham.ac.uk/~jbb|~aps|~vxs

Masakazu Suzuki
Faculty of Mathematics
Kyushu University, Japan
suzuki@math.kyushu-u.ac.jp
http://www.math.kyushu-u.ac.jp/~suzuki/

***Abstract*—Document analysis of mathematical texts is a challenging problem even for born-digital documents in standard formats. We present alternative approaches addressing this problem in the context of PDF documents. One uses an OCR approach for character recognition together with a virtual link network for structural analysis. The other uses direct extraction of symbol information from the PDF file with a two stage parser to extract layout and expression structures. With reference to ground truth data, we compare the effectiveness and accuracy of the two techniques quantitatively with respect to character identification and structural analysis of mathematical expressions and qualitatively with respect to layout analysis.**

***Keywords*-Math formula recognition, PDF, layout analysis**

I. Introduction

The correct recognition of mathematical texts is a challenging problem for document analysis and a significant amount of research has been devoted to the topic [1]. With some exceptions (cf. [2]), most of this work has concentrated on recognising mathematics from scanned images of print or handwriting. However, much of the modern mathematical literature, in particular research papers and journal articles, is available in PDF, for which there is very little support for extracting the mathematics contained therein. Even simple tasks like copying and pasting expressions are often unavailable, let alone more advanced processing like translating to a suitable markup language such as LaTeX or MathML or making it accessible to screen readers. While PDF viewers can perform these task for ordinary text, and conversion tools can translate it into other input formats like Word or LaTeX, they fail on embedded mathematics. Thus document analysis for mathematical texts in PDF is still a challenging problem.

Two major approaches deal with this problem. One, implemented in the Infty system, reduces it to the traditional approach for scanned documents. It uses OCR for symbol recognition and a virtual link network for structural analysis. Whilst it makes full use of sophisticated mathematical formula recognition techniques, it ignores the additional information that is contained in the PDF documents. The second approach exploits this information by directly extracting symbol information from the PDF file and uses a two stage parser to analyse layout and mathematical expression structures. This is implemented using bespoke PDF extraction tools for mathematical documents [3]. These tools, originally developed to work exclusively on single, possibly multiline mathematical expressions pre-identified in a PDF document, have been extended to work on entire pages of documents by adding support for layout analysis and segmentation of mathematics from ordinary text.

While we present the novel additions to the latter approach here, we consider the main contribution of this paper to be the experimental comparison between the two alternative approaches. This is carried out with respect to a ground truth set that has been especially constructed for this purpose. It is modelled on that constructed for the Infty system [4], but exclusively uses documents in a suitable PDF format. As performance metrics we employ character and symbol recognition rate as well as structure recognition rate as identified in [5]. This is supplemented by a more subjective comparison of the similarity of the rendered LaTeX output to the original document as produced by both approaches.

II. OCR-based Analysis of PDF Documents

The first approach we present is primarily based on standard OCR. This is implemented in the Infty system [7] and uses that system's facilities for recognising mathematical texts from scanned documents. That is, it first renders the PDF document into an image before completing layout analysis, segmentation and character and mathematical expression recognition. High recognition rates can be obtained with digital born PDF documents as they are relatively free of noise, thus less prone to common recognition errors.

A. OCR and Math Segmentation

When presented with a scanned or retro-digitised document, some standard prepreocessing steps are completed in order to make the document suitable for OCR and document analysis: namely binarisation, noise removal and deskewing followed by the extraction of connected components.

Analysis, mainly of the connected components' sizes, is then used to segment the document into areas of figures and non-figures. Large connected components are generally treated as figures, with special rules to segment large mathematical characters such as root symbols and big parentheses. Further analysis of the-non figure area is then completed to identify individual lines. In non-mathematical documents, this is a trivial task, with line boundaries identified via unbroken horizontal white space. However this is not suitable for mathematics, and a technique similar to that described by Kacem et al [8] is used, where components in certain areas are concatenated to build lines.

Non-figure areas are initially passed through a commercial OCR system, which, in conjunction with a large lexicon, produces high quality recognition results on standard text, but fails when presented with the various fonts, irregular baselines and unusual symbols found in mathematics, producing meaningless strings with low confidence. These areas of high misrecognition are marked as mathematics.

The second stage of segmentation involves overlaying the OCR results onto the original document. The bounding box position and size of each recognised character is compared with the original. If they vary above a certain threshold, the characters are also treated as mathematics. This method is particularly effective when there are changes in baseline, so can identify when expressions containing sub and superscripts have been misrecognised as text.

The connected components identified as part of mathematical expressions are then subject to an OCR engine designed for symbols commonly found in mathematics, such as Greek and Latin letters, numerals, operators, large symbols, etc. This engine is also designed to recognise font types and styles, such as italic, bold, roman and calligraphic.

The recognition results, for both the math and standard text, are then subjected to supervised clustering in order to improve results. The connected components are grouped into similar shapes and put into sets based upon character categories. Majority voting is carried out within each cluster to decide whether a component can stay in that cluster.

B. *Formula recognition*

Infty use a virtual link network for formula recognition, which constructs a network of vertices and edges with costs representing the characters and their spatial relationships. Each vertex within the network consists of a number of candidate characters with costs based upon the results from the OCR software. Directed edges are created between pairs of characters, representing the relationship between a parent and its child(ren). Costs are assigned to each edge based upon the type of link and the distances between the normalised centres, sizes and types of the characters.

Possible relationships between a child c and parent P are: Horizontal Pc, right superscript P^c, right subscript P_c, left superscript ${}^cP$, left subscript ${}_cP$, under $\overset{P}{c}$, and over $\overset{c}{P}$. Edges are only created when they fulfil certain criteria, such as scripts being smaller than parents and a line of sight existing.

Admissible spanning trees based upon these link networks are then searched for, and output if they have a total cost below a predetermined level. An admissible spanning tree has to meet the following four criteria:

1) Each node has a maximum of 1 child with the same label.
2) Each node has a unique candidate chosen by its linked edges.
3) The super or subscript sub tree to the right of a node K is left of the horizontally adjacent child of K.
4) The super or subscript sub tree to the left of a node K is right of the horizontally adjacent parent of K.

Once the list of admissible candidate trees is created, their costs are reevaluated, adding penalties if certain conditions are met. These conditions are generally based around unusual relationships between nodes. Once this final step has been completed the tree with the lowest cost is returned.

C. *Layout Analysis*

A text area, i.e. a non-figure and non-mathematical area, once divided into lines can be further split into specific elements via the analysis of the size, spacing, case, typeface and positioning of characters within the line, together with the identification of keywords. Structural elements that Infty can identify are; titles, headings, author information, headers, footnotes, page numbers and mathematical components.

- Titles are identified by appearing in the upper part of the first page and in a large font.
- Headings are identified by consisting of larger than average fonts, beginning with numbers, or containing keywords such as *introduction*, or *bibliography*. The size of the characters is used to distinguish major and minor headings.
- Text areas after the title are marked as author information.
- Headers and footers are identified by having smaller than average sizes and appearing at the top or bottom of the page.
- Page numbers are identified by numerals appearing at the bottom or top left or right of a page.
- Mathematical components are indicated by keywords, such as 'Lemma' and 'Theorem', their styles and the styles of the following text.

III. Extraction of PDF Information

While the previous approach has the advantage that the OCR engine is applied to a noise free image produced from the PDF version, it fails to use any of the information available from the PDF document. We now present an approach that aims to do exactly this, by extracting information on characters, their fonts and sizes, together with their exact position in the document. This information is then exploited to reassemble the original document with a particular emphasis on the mathematical expressions. These techniques are implemented in the specialist PDF extraction tool Maxtract, that uses a linear grammar approach for recognising mathematical expressions [3] together with font and size information on characters for augmented recognition [9]. For this comparison the tool was substantially extended to not only work on manually clipped mathematical expressions alone but automatically on entire PDF documents, adding layout analysis and segmentation of mathematics and text.

A. PDF extraction

Initially, all characters on a given page together with their exact positioning have to be extracted. Unfortunately, PDF documents do not contain the true bounding box information about the characters that they contain. Instead, they specify the point where the characters are rendered to on the page and provide only a very crude bounding box approximation for each character. Whilst this information is adequate for analysing standard text on a single baseline, it is insufficient for mathematical expressions, where characters are not only arranged two-dimensionally but can also vary drastically in size. Naive use of this information from PDF files can lead, not only to imprecise, but to fundamentally incorrect results about actual character placement.

To obtain the precise bounds necessary, the PDF document is rendered to a 600dpi TIFF bitmap image, with the bounding boxes of each glyph (i.e., connected component) in the page extracted then registered with the character information from the original PDF. For most characters this is a simple matter of translating and scaling the coordinates obtained from the PDF and matching them to the corresponding glyphs. However, the task is considerably complicated by the fact that we have to cater for those cases where (a) single characters consist of multiple glyphs, e.g. "$=$" (b) single glyphs correspond to multiple PDF characters, e.g. large fences or root symbols, and (c) some multiple glyphs correspond to multiple characters in complex ways. The extraction produces a precise list of characters on the PDF page, their bounding boxes and font and size information.

In our previous work, this information was used solely for the structural analysis of single, manually identified mathematical expressions [3], [9]. However, for this comparison Maxtract was extended to process entire pages. As a consequence, layout analysis was required in order to identify lines and columns. This was completed by searching for unbroken vertical white space between horizontally ordered characters to identify columns, and unbroken horizontal white space between vertically ordered characters in columns to identify lines. An analysed PDF page consists of a number of lines, each with an overall bounding box and list of symbols, with character and glyph information.

B. Linearisation of Expressions

Each extracted line is then parsed separately to reconstruct its spatial layout. In a first parsing step characters are clustered exploiting both the extracted information on their size and font as well as spacing information computed from their bounding boxes. Thereby the system employs a simple thresholding approach based upon character widths to categorise the white space between characters into five different classes $0, \ldots, 4$. Characters are grouped together if they are either alpha characters or single digits and they (a) use the same font, (b) have the same font size, (c) have the same base y coordinate, i.e. they share a baseline, and (d) the space between the two nearest edges of any adjacent pair is in threshold class 0. This yields single characters or groups of characters, which form words or numbers.

The second step proceeds to linearise the 2-dimensional layout of the characters into a 1-dimensional version using rules of mathematical expression composition. The rules are based on an original set given by Anderson [10]. However, the rule set has been significantly extended to cover a far wider range of mathematics. Also, existing rules were heavily modified in order to cope with various styles of typeset notation. The grammar contains 12 rules to deal with the different spatial relationships between sets of symbols including scripts, fractions, limits, contained symbols, matrices, cases, accents and symbols spread over multiple lines. The intention here is not yet to do a full parsing of the mathematical formula, but rather a spatial analysis of the formula, separating it into one dimensional segments of characters that sequentially follow each other on the same baseline. For example, omitting representational details, "$f(x)e^{-i\omega t}$" would be linearised into "$\texttt{sup}(\ f(x)e\ )\ (\ -i\omega t\ )$"

C. Layout Analysis and Math Segmentation

Given the line by line information, the layout analysis proceeds in two steps. First, lines are separated into text lines and display style mathematics, which are then grouped together into paragraphs and further classified.

The separation of text and math lines is based both on spatial positioning of the line with respect to the left and right margin of the page as well as the number of words on that line, where a word is a sequence of alpha characters, grouped by the previous parsing step. A line is then treated as a text line if it (a) contains only a sequence of words, (b) contains at least two consecutive words and the number of other expressions is not larger than the number of words, (c) contains more than three consecutive words regardless of the number of other expressions. Everything else will be treated as display style mathematics.

In the second step consecutive display style math lines are combined into single multi-line math expressions, which can be further exploited in the structural analysis described in the next section. Consecutive text lines are combined into paragraphs, where paragraphs are separated if (a) there is a change of font size, (b) the vertical space between lines is larger than the arithmetic median of vertical space between all consecutive text lines identified on the page, (c) the horizontal orientation of lines changes, (d) if a line has a left indentation or the previous line ends prematurely. After paragraphs have been grouped, an attempt is made to identify special text areas such as page numbers or headings. For example the latter are identified as single lines in which all characters have either a different font or a larger font size than the majority of characters on the page.

	[11]	[12]	[13]	[14]	[15]
Objects	11143	3233	1935	2418	2120
Misrecognised	53	5	5	1	3
Extras	46	2	6	2	3
Missing	10	5	4	0	5

Table I
INFTY CHARACTER RECOGNITION RESULTS

D. *Structural Analysis*

The 1-dimensional linearised lines are parsed using a LALR parser, resulting in a parse tree that is used as an intermediate representation for subsequent translation into various output formats. Structural information in the parse tree can be exploited by these translation modules.

For example, when translating the parse tree into LaTeX output, spatial information is used in multi-line math expressions for alignment purposes. Text lines are translated by linearly assembling consecutive words, identifying sequences of mathematical expressions and assembling them into single inline math formulae. Common interpolation symbols (commas, periods etc.) are always attached to the previous element, either to a word or a math expression. In addition the contained font and spacing information is exploited to set characters and words in the correct font and size as well as to include additional space if necessary.

IV. EXPERIMENTAL COMPARISON

A. *Character Recognition Rate*

To determine the accuracy of the Infty character recognition results, they were compared to a manually verified and corrected ground truth set, with four areas for comparison identified shown in table I.

Objects The total number of objects identified, including characters, lines and control structures such as spaces.

Misrecognised The number of objects with a corresponding object in the ground truth set, but a differing character code. E.g., when a $\top$ is incorrectly recognised as a T.

Extras The number of objects identified without a corresponding object in the ground truth set. This can occur when additional spaces are identified, or symbols are split into more than one character, such as the ligature fi being recognised as the separate characters f and i.

Missing The number of objects in the ground truth set without a corresponding object in the results. This can occur if spaces are missed, punctuation is removed as noise or glyphs are incorrectly joined together into a single character

The worst recognition rates were found with [11], in particular caused by the tight spacing, unusual symbols and non standard fonts. Mistakes included $\diamond$ being recognised as O, $\top$ as T, an unrecognised Fraktur symbol and a number of spacing errors. Throughout the other documents the main errors were incorrect cases and spacing errors.

The accuracy of Maxtract was determined by identifying any unmatched glyphs or PDF characters after the matching phase and identifying any symbols that could not be reproduced in LaTeX, with the results shown in table II

	[11]	[12]	[13]	[14]	[15]
Characters	9304	2799	1744	2094	1889
Symbols	9282	2785	1729	2094	1868
Misrecognised	0	0	0	0	0
Missing	0	0	0	0	0

Table II
MAXTRACT CHARACTER RECOGNITION RESULTS

	Infty	Maxtract		Infty	Maxtract
Expr found	635	850	Add. chars	10	102
Correct	550	235	Misrecogn.	33	16
Expr split	40	172	Not recogn.	7	0
Space diffs	2	103			

Table III
STRUCTURE RECOGNITION RATE WRT. 628 EXPRESSION.

Characters The total number of lines and PDF characters extracted from the PDF file. This can include standard characters such as a, 1 or =, and characters which form part of a larger symbol. Some symbols, especially large ones, are made from multiple characters and lines. For example, a large bracket can be constructed of $\lceil$, $\lfloor$ and multiple $|$.

Symbols The number of symbols identified, after mapping characters to glyphs. This is often less than the number of characters extracted, due to multi-character symbols.

Misrecognised The number of symbols that cannot be converted to Unicode. They occur when character names are incorrect or missing from the font encoding of the PDF.

Missing The number of orphan characters remaining after glyph matching.

No character recognition errors occurred using Maxtract.

B. *Structure Recognition Rate*

The ground truth set contained 628 separate mathematical expressions in our sample articles. The comparison of the structure recognition rate of the two approaches is given in table III. One can observe that the number of expressions found by Maxtract is significantly larger, which is primarily due to the fact that Maxtract classifies every non-alpha character it finds as math — including all digits or brackets around citations — while this is not the case for Infty or in the ground truth. Furthermore, explicit spaces are always designated as non-math, which leads to a number of single expressions in the ground truth recognised as several expressions separated by white space in Maxtract. To a lesser degree the same phenomenon can be observed for Infty's recognition result. The difference in recognition of explicit white space between Infty and Maxtract also leads to the discrepancy of figures in row 4, where the low number for Infty is not surprising given the fact that the ground truth set has been constructed by manually correcting Infty's original recognition results. A similar argument can be used to explain the figures for "additional characters"

Original	Infty	Maxtract
$r \in \mathsf{N}_{\mathsf{glo}}.$	$r \in \mathrm{N}_{\mathrm{g}^{1\circ}}.$	$r \in \mathsf{N}\ _{\mathsf{glo}}\ .$
$\mathfrak{I}$	$\sim /!$	$\mathfrak{I}$
$\sum_{i=0}^{m} a_i x^i,$	$\sum_{i=0}^{m} a_i x^i$	$\overset{m}{} \sum a_i x^i, \quad i = 0$

Table IV
COMPARISON OF RENDERED LaTeX RESULTS

where Maxtract tends to add more commas or periods at the end of mathematical expressions.

Thus we can view the figures on misrecognised expressions as the true failure in the structure recognition. Of the 33 expression Infty misrecognises, over 20 are from the Artale paper, and can be attributed to problems with the fonts and squeezed space in this paper. All the remaining mistakes are misrecognised or incorrect extra subscripts. From the results for Maxtract, 8 are differences in recognition of stacked expressions, that are recognised as "Over" nodes in the ground truth, but as "Under" nodes by Maxtract. A further 2 are due to a difference in recognition of case splits. Of the remaining 6 recognition errors, one is a true misrecognised subscript whereas the other 5 are down to Maxtract recognising a convolution operator $\hat{}$ as an accent rather than a superscript. Finally, Maxtract recognises all mathematical expressions that are also in the ground truth, whereas Infty fails to recognise 7 (3 in Artale).

C. LaTeX Comparison

The first two examples in table IV shows the difficulty that traditional OCR systems such as Infty face when dealing with the large variety of fonts and characters commonly found in mathematics. Because of a varying baseline in the first example along with a non standard font, the l and o are recognised as 1 and $\circ$ respectively. The second example shows a Fraktur I being recognised as multiple characters, likely to be caused by the fact that it is constructed from more than one glyph. Maxtract did not have any problems recognising these symbols.

The third example show Maxtract producing incorrect output for a summation. This is because the formula has been incorrectly segmented into three lines, due to the unbroken horizontal white space existing between the limits and the main body. Due to the more advanced line finding algorithm, this is not a problem for Infty.

V. CONCLUSIONS

When presented with a suitable PDF file, Maxtract produces perfect character recognition results and can produce a high quality reconstruction of text and formulae in LaTeX, with parse trees of formulae with semantic information. However the approach to line and math segmentation is somewhat naive and can incorrectly split a single formula into multiple lines. A more refined approach, such as the n-gram based methods described in [6] may be more effective.

Infty produces very high quality OCR results from born-digital PDF and works well on a wide range of documents. It also boasts very good layout analysis and the ability to extract the logical structure of a document. However, it can not accurately produce the same depth of character and font information as Maxtract and its formula recognition is more focused on spatial layout rather than semantic analysis.

Infty is a fully integrated end to end system and Maxtract is a tool that works specifically with suitable PDF documents. Our results show that the two systems complement each other well and integrating Maxtract into Infty would lead to a system with perfect character recognition on PDF files, advanced layout analysis and the ability to reconstruct documents with high quality semantic and syntactic mark up.

ACKNOWLEDGMENT

We thank the Royal Society for their support through the International Joint Project 2008/R3.

REFERENCES

[1] K. Chan and D. Yeung, "Mathematical expression recognition: a survey," *IJDAR*, vol. 3, pp. 3–15, 2000.

[2] M. Yang and R. Fateman, "Extracting mathematical expressions from postscript documents," in *Proc. of ISSAC '04*. ACM Press, 2004, pp. 305–311.

[3] J. Baker, A. Sexton, and V. Sorge, "A linear grammar approach to mathematical formula recognition from PDF," in *Proc. of Intelligent Computer Mathematics*, 2009.

[4] M. Suzuki, S. Uchida, and A. Nomura, "A ground-truthed mathematical character and symbol image database," in *Proc. of ICDAR*. IEEE Computer Society, 2005, pp. 675–679.

[5] A. Lapointe and D. Blostein, "Issues in performance evaluation: A case study of math recognition," in *ICDAR*. IEEE Computer Society, 2009, pp. 1355–1359.

[6] U. Garain, "Identification of Mathematical Expressions in Document Images," in *ICDAR*. IEEE Computer Society, 2009, pp. 1340–1344.

[7] M. Suzuki, F. Tamari, R. Fukuda, S. Uchida, and T. Kanahori, "Infty — an integrated OCR system for mathematical documents," in *Proc. of DocEng*. ACM Press, 2003, pp. 95–104.

[8] A. Kacem, A. Belad, and M. Ben Ahmed, "Automatic extraction of printed mathematical formulas using fuzzy logic and propagation of context," *IJDAR*, vol. 4, pp. 97–108, 2001.

[9] J. Baker, A. Sexton, and V. Sorge, "Faithful mathematical formula recognition from PDF documents," in *Proc. of 9th DAS,*. ACM Press, 2010, pp. 485–492.

[10] R. H. Anderson, "Syntax-directed recognition of hand-printed two-dimensional mathematics," Ph.D., Harvard, 1968.

[11] A. Artale, C. Lutz, and D. Toman, "A description logic of change," *Proc. of IJCAI-20*. Morgan Kaufmann, 2007.

[12] R. Durrett, *Probability: Theory and Examples*, 4th ed. Cambridge University Press, 2010,

[13] T. W. Judson, *Abstract Algebra — Theory and Applications*, 2009. [Online]. http://abstract.ups.edu/download.html

[14] D. R. Wilkins, "On the number of prime numbers less than a given quantity," 1998. [Online]. http://www.maths.tcd.ie/pub/HistMath/People/Riemann/Zeta/EZeta.pdf

[15] S. Sternberg, "Theory of functions of a real variable," 2005. [Online]. http://www.math.harvard.edu/~shlomo/docs/Real_Variables.pdf