Automatic Extraction of Figures from Scholarly Documents

Sagnik Ray Choudhury
Information Sciences and Technology
Pennsylvania State University
sagnik@psu.edu

Prasenjit Mitra
Information Sciences and Technology
Pennsylvania State University
pmitra@ist.psu.edu

Clyde Lee Giles
Information Sciences and Technology
Pennsylvania State University
giles@ist.psu.edu

ABSTRACT

Scholarly papers (journal and conference papers, technical reports, etc.) usually contain multiple "figures" such as plots, flow charts and other images which are generated manually to symbolically represent and illustrate visually important concepts, findings and results. These figures can be analyzed for automated data extraction or semantic analysis. Surprisingly, large scale automated extraction of such figures from PDF documents has received little attention. Here we discuss the challenges of how to build a heuristic independent trainable model for such an extraction task and how to extract figures at scale. Motivated by recent developments in table extraction, we define three new evaluation metrics: figure-precision, figure-recall, and figure-F1-score. Our dataset consists of a sample of 200 PDFs, randomly collected from five million scholarly PDFs and manually tagged for 180 figure locations. Initial results from our work demonstrate an accuracy greater than 80%.

Categories and Subject Descriptors

H.4 [**Document Analysis**]: Information Extraction; D.2.8 [**PDF processing**]: machine learning

General Terms

figure extraction; machine learning; PDF processing

Keywords

figure extraction; PDF; document analysis

1. INTRODUCTION

Scholarly papers often contain multiple "figures" some of which are generated from data which is not reported anywhere else in the paper, making them invaluable sources of information, available only there. These figures can be manually extracted from PDF documents using free software such as Inkscape. However, a batch extractor is necessary for extraction at scale, say for a search engine. Previous research explored automatic classification and data extraction from specific types of figures, mostly line graphs and scatter plots [2, 11](see section 2). But, these methods have not yet been applied to large scale datasets.

One possible way to extract such figures is to segment a page image (a PDF page converted into an image or a scanned image of a document page) into text and graphics region. This is a well-known research problem dating back to the 1980s [13]. These algorithms assume a considerable difference in pixel densities in the text and graphics regions. While that is true for most newspaper/magazine documents, engineering drawings, and scholarly papers are different [3].

There is an abundance of "born digital" scholarly documents, mostly in the PDF format. It is certainly possible to convert a born digital PDF document into page images and apply existing image segmentation algorithms [4, 14]. But, PDF to image conversion can be computationally expensive. Our experiments suggest that the average CPU time for PDF page to image conversion is more than two seconds, while our PDF processing-based approach takes half of that.

Figures are embedded in PDF documents in raster (PNG, JPEG) or vector formats (SVG, EPS). Typically, raster images are embedded in the PDF as separate content streams (XObjects). Therefore, it is easy for a PDF parser to extract them but it is hard to extract vector graphics as PDF documents themselves are written in the same format. In a PDF document, graphics and textual elements are often interleaved in the content stream and can not be easily segregated. Previous work by Futrelle et al. [8], Shao et al. [15] and very recently, Clark et al. [7] have reported heuristic methods for this task. But as these methods depend strongly on heuristics, they are often hard to reproduce since they require elaborate manual tuning. More importantly, most previous work has not discussed any strategy or results for the evaluation the extraction accuracy (with the exception of Clark et al. [7]).

We report a machine learning based method that does not depend on heuristics to extract figures from PDF documents. We also define metrics to evaluate the extraction accuracy and create a tagged dataset for use and future evaluation. We show that our method is more scalable with regards to existing approaches. Plus, our average F1-score is higher than 80%.

2. RELATED WORK

In earlier work, we described machine learning based algorithms for figure metadata extraction [5] and a search engine

DocEng'15, September 8-11, 2015, Lausanne, Switzerland.

ACM 978-1-4503-3307-8/15/09 ...$15.00.
DOI: http://dx.doi.org/10.1145/2682571.2797085.

on the extracted metadata [6]. Here, we focus on figure extraction, especially vector graphics.

This problem can be solved in a two-step process: 1. Convert a PDF page into an image, and 2. Segment that image into text and graphics regions. There has been considerable work in such page segmentation. The approaches can be broadly classified into three classes: 1. A top-down block based approach where the image is segmented into blocks, and these blocks are classified as text or graphics region [13]; 2. A bottom up pixel-based approach where each pixel is classified as image or text and finally grouped together to form larger blocks [16]; and 3. A morphological operation based approach where a "graphics mask" is created through a set of morphological operations [1]. As discussed in section 1, these approaches suffer from a scalability problem. Also, these methods assume that the pixel density of a figure region is significantly different from that of a text region, which is usually not true for scholarly documents [3]. Our previous work [14] improved the morphological segmentation but suffered from scalability.

Little work([7, 8, 15]) has explored figure extraction from PDF documents by processing the PDF primitives. As mentioned before, they suffer from problems such as manual tuning of heuristics and lack of standardized evaluation metrics and data sets.

3. USING PDF OBJECT MODEL FOR FIGURE EXTRACTION

At an abstract level, a PDF document can be described using three types of primitives: text, path (vector elements such as lines and curves) and raster (bitmap) images. Each primitive is painted on the screen using a set of operations, the graphics state, and a transformation matrix.

We output bounding boxes for figure regions on a page merging paths and bitmap images. These bounding boxes contain the text inside the figure.

Extraction of bitmap images is relatively easy because there are only four operators for painting the image, and the painting locations can be extracted easily as well. The scenario is more complicated For the vector elements. Three types of operators are used to render a path:

1. **Path construction**: Operators c (curveto), l (lineto) and others are used to define start and end points of a path.

2. **Path painting**: Operators S (stroke path), s (close and stroke path) are used to paint a path on the screen. These operators determine the color, width and other esthetic details.

3. **Clipping paths**: Operators W and W^* are used to construct clipping paths. A clipping path defines a region of the page which should be used for painting the vector elements.

Extracting the locations for "paths" can be difficult because the parser needs to consider various elements of the graphics environment, as described before. Therefore, it is beneficial to have a higher level representation of the PDF. Recent work by Hassan et al. [10] introduced an "object level" representation of PDF documents. Their software (pdfXtk) produces bounding boxes of vector and raster graphic elements from a PDF document, combining several small subpaths. The goal of the software is to "obtain a simplified representation of the most important lines and boxes which are of material importance for layout analysis, i.e. they are likely to be noticed immediately by a human reader just scanning through the page and are at the level of granularity required for performing document analysis" [10]. Our system uses the output of this software.

The output from pdfXtk are the bounding boxes for paths, but not all of them belong to graphics regions. For example, most tables have lines, symbols can be drawn by curves. These paths need to be filtered out before the grouping. Surprisingly, previous works don't discuss that. We propose models learned from the data to remove these "noisy paths". Also, we show that the grouping can be done using clustering algorithms, removing the need for heuristics. Once paths are classified and clustered, their bounding boxes can be merged to produce final figure regions.

3.1 Classification of Raster Graphics and Paths

We considered a binary classification problem where we classified each path/raster graphic as a member of a figure region (positive) or not (negative). We observed that most large raster graphics (area of the graphic $> 10\%$ of the page area) belonged to the positive class. Paths pose a greater challenge and determining such heuristic is hard. We extracted following features for each path:

- **Character density ratio**: It is intuitive that the paths inside the figure regions will have less text around them whereas paths inside table/equation region will have a higher amount of text. Therefore, for each path, we extract the character density within a region around it. Character density is defined as the number of characters inside a region / area of the region. We also extract the character density for the whole document. The feature value is defined as CD_p/CD_{pdf}(character density of the path / character density of the PDF).

- **Distance from boundary**: PDF pages usually contain paths which acts as demarcations or used for decorative purposes. For example, footnotes are usually separated from the main content by a straight line near the boundary of the page. On the contrary, paths inside figure regions are far from the page boundary. We define the distance of a path from a boundary as the minimum of distances from all axes.

- **Number of paths in ϵ neighborhood**: This feature is motivated from DBSCAN algorithm where a point is classified as noise if it has less than N points in its ϵ neighborhood. Often, paths are used to paint symbols such as ratio (/), summation (Σ). As these paths should be inside the text regions and not the figure regions, ideally they should have less number of paths around them.

- **Area**: We observed that the paths inside the figure regions had smaller area compared to the other paths.

We experimented with four classifiers, and the results are reported in section 4.3.

3.2 Combining Paths into Figure Regions

Paths classified as positive instances can be grouped by heuristics to create figure regions. A popular way for such grouping is optimized X-Y cut, [12] but the parameters for the algorithm need to be tuned. Therefore, our system uses a clustering algorithm.

A clustering algorithm such as K-means has three parameters: 1. The number of clusters, 2. Distance function and 3.

Initialization. Most scholarly documents contain figure captions. Therefore, number of clusters can be estimated easily using regular expressions. From our previous work [14], it was evident that the best distance function is the Euclidean distance between the centers of the bounding boxes. In the usual implementations of K-means algorithm, the initial points are chosen randomly, which can lead to arbitrary results. We experimented with two initialization methods:

- **Nearest point to a figure caption (NFC)**: Points nearest to the figure captions were used as initialization points. The distance is measured by Manhattan distance between two rectangles.

- **K-means++**: In this method, cluster centers are chosen to be far away from each other. The first initial cluster center is chosen at random. The second cluster center x is chosen with a probability proportional to the distance of the point x from the first cluster center. This process is repeated until K cluster centers are chosen.

4. EXPERIMENTS AND RESULTS

4.1 Dataset

We randomly sampled 200 PDF files from CiteSeerX repository and split them into pages, yielding approximately 1800 pages. 85 pages each having more than one figure and more than five paths/images were randomly selected as test data.

From the rest, we randomly selected 50 pages containing at least one figure to generate the data for the classification experiment. We extracted approximately 3000 paths, but more than 85% of these paths belonged to the positive class. This is not surprising, given that most paths would belong to some figure. However, this could create highly overfitted models, especially in decision trees. To solve this data imbalance problem, we further sampled 50 pages that contained no figure but tables. The final data for classification (approximately 4000 paths) had 2:1 positive to negative ratio.

Since figure regions had to be manually tagged from the page images, and we only investigated the harder cases (pages with minimum two figures), the dataset is relatively small. A completely random selection would include pages with one figure. Though that would increase the accuracy, there would be no clustering evaluation.

4.2 Evaluation

For the classification problem, we use well-known evaluation metrics: precision, recall, and F1-score. For our problem, it is important to have high recall for the negative class, even at the expense of the positive class. Because, even if some positive samples belonging to a figure region are wrongly classified, they would possibly be merged into the region due to the correctly classified samples. Figure 1 shows an example of that case.

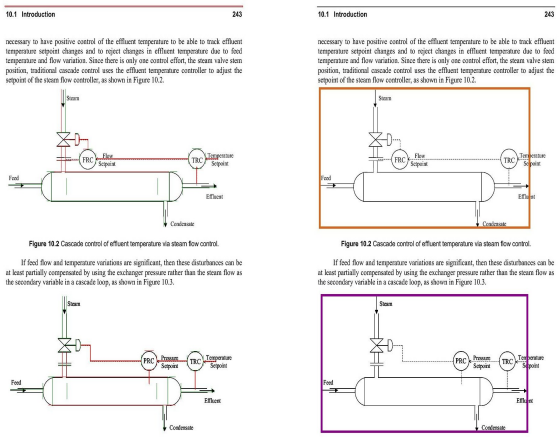
The clustering evaluation is tricky. Standard metrics such as adjusted rand index are not suitable because even a single point clustered wrongly can change the region size dramatically. Suppose the location of the actual figure is given by a rectangle R_g and the predicted location is given by a rectangle R_p. The evaluation metrics are defined as:

1. **Figure-precison**: $\frac{\text{Area overlap between } R_g \text{ and } R_p}{\text{Area of } R_p}$

2. **Figure-recall**: $\frac{\text{Area overlap between } R_g \text{ and } R_p}{\text{Area of } R_g}$

3. **Figure-F1-score**:Harmonic mean of figure-precison and figure-recall.

Given the gold standard data for a page (i.e. set of all R_gs for that page) and the predicted locations for the same page (set of all R_ps for that page) we first calculate the correspondence between the sets. A correspondence configuration is defined as a one to one mapping between two sets. We calculate the total area overlap for all such possible configurations and the one having the maximum value is considered to be the final mapping. Once the mapping is defined, we calculate the "figure-precision", "figure-recall" and "figure-F1-score" values between (R_p, R_g) pairs. Our metrics are motivated by ICDAR 2013 table localization competition [9].



(a) Classification results on a sample page.

(b) Clustering results on the classified paths.

Figure 1: An example where some instances of the positive class (green) are classified as negative class (red). However, that doesn't change the clustering quality.

4.3 Classification: Results and Discussions

We experimented with four classifiers for the classification problem: 1. A Linear Kernel SVM (penalty parameter value=1), 2. A Gaussian Naïve Bayes classifier, 3. A Decision Tree classifier with depth=3 and Gini index as the splitting criterion and 4. A Logistic Regression classifier. The data (4000 paths) was splitted in 70:30 ratio for training and testing, maintaining the class balance. Each classifier was run 200 times, and the metrics were calculated on the test data. The results of the experiments are presented in table 1. Note that the experiment process is equivalent to stratified cross-validation but more robust as it is done for 200 times. The decision tree performed better in classifying the negative class. More importantly, the recall is the highest when we use decision trees. Inference in decision trees is rule-based, hence scalable. We experimented with multiple combinations of the features, but the results didn't improve.

4.4 Clustering: Results and Discussions

We used the decision tree model learned from the training data to classify each path in test data. Positively classified paths were clustered using K-means, and finally merged into figure regions. For the clustering problem, we experimented

Classifier	Recall		Precision		F1-Score	
	(-)ve	(+)ve	(-)ve	(+)ve	(-)ve	(+)ve
SVM	53.9	87.7	65.4	82.0	58.7	84.7
Naïve Bayes	45.5	92.0	70.4	80.1	55.1	**85.7**
Decision Tree	73.2	73.1	53.7	86.8	**61.7**	79.1
Logistic Regression	72.4	69.1	49.5	85.7	58.8	76.5

Table 1: Classification results for used classifiers.

with two initialization parameters. The results are presented in table 2. For the first method of initialization (Nearest point to a figure caption), clustering is deterministic because the choice of the initial cluster centers is deterministic. For the second method (K-means++), the choice is probabilistic. Therefore, we ran the clustering process ten times and chose the fifth output. We ran the clustering and merging process on 85 pages, each having more than one figure region and five paths. Figure precision, recall, and F1-scores were calculated as described in section 4.2. We had 180 figures in the gold standard. Table 2 presents the average values for the metrics. As expected, the first initialization method outperforms the K-means++ method.

Initialization method	Figure-precison	Figure-recall	Figure-F1-score
Nearest point to a figure caption	81.9	85.0	80.9
K-means++	78.4	80.4	76.6

Table 2: Figure-precison,recall and F1-scores on test data.

5. CONCLUSION AND FUTURE WORK

We propose a machine learning based approach to extract figures from scholarly PDF documents. Our system builds on recent developments in document processing. Contrary to most work in this area, our approach is heuristic independent and achieves good accuracy and scalability. We have also designed evaluation metrics and created a labeled dataset. Future work would be to improve the clustering algorithm and explore the table extraction problem using a similar approach.

6. ACKNOWLEDGEMENTS

We gratefully acknowledge partial support from the National Science Foundation and NPRP grant # 4-029-1-007 from the Qatar National Research Fund (a member of Qatar Foundation).

7. REFERENCES

[1] D. S. Bloomberg. Multiresolution morphological approach to document image analysis. In *Proc. of the International Conference on Document Analysis and Recognition, Saint-Malo, France*, 1991.

[2] W. Browuer, S. Kataria, S. Das, P. Mitra, and C. L. Giles. Segregating and extracting overlapping data points in two-dimensional plots. In *Proceedings of the 8th ACM/IEEE-CS joint conference on Digital libraries*, JCDL '08, pages 276–279, New York, NY, USA, 2008. ACM.

[3] S. S. Bukhari, F. Shafait, and T. M. Breuel. Improved document image segmentation algorithm using multiresolution morphology. In *IS&T/SPIE Electronic Imaging*, pages 78740D–78740D. International Society for Optics and Photonics, 2011.

[4] H. Chao and J. Fan. Layout and content extraction for pdf documents. In *Document Analysis Systems VI*, pages 213–224. Springer, 2004.

[5] S. R. Choudhury, P. Mitra, A. Kirk, S. Szep, D. Pellegrino, S. Jones, and C. L. Giles. Figure metadata extraction from digital documents. In *Document Analysis and Recognition (ICDAR), 2013 12th International Conference on*, pages 135–139. IEEE, 2013.

[6] S. R. Choudhury, S. Tuarob, P. Mitra, L. Rokach, A. Kirk, S. Szep, D. Pellegrino, S. Jones, and C. L. Giles. A figure search engine architecture for a chemistry digital library. In *Proceedings of the 13th ACM/IEEE-CS joint conference on Digital libraries*, pages 369–370. ACM, 2013.

[7] C. Clark and S. Divvala. Looking beyond text: Extracting figures, tables and captions from computer science papers. In *Workshops at the Twenty-Ninth AAAI Conference on Artificial Intelligence*, 2015.

[8] R. P. Futrelle, M. Shao, C. Cieslik, and A. E. Grimes. Extraction, layout analysis and classification of diagrams in pdf documents. In *2013 12th International Conference on Document Analysis and Recognition*, volume 2, pages 1007–1007. IEEE Computer Society, 2003.

[9] M. Gobel, T. Hassan, E. Oro, and G. Orsi. Icdar 2013 table competition. In *Document Analysis and Recognition (ICDAR), 2013 12th International Conference on*, pages 1449–1453. IEEE, 2013.

[10] T. Hassan. Object-level document analysis of pdf files. In *Proceedings of the 9th ACM symposium on Document engineering*, pages 47–55. ACM, 2009.

[11] X. Lu, S. Kataria, W. J. Brouwer, J. Z. Wang, P. Mitra, and C. L. Giles. Automated analysis of images in documents for intelligent document search. *IJDAR*, 12(2):65–81, 2009.

[12] J.-L. Meunier. Optimized xy-cut for determining a page reading order. In *ICDAR*, volume 5, pages 347–351, 2005.

[13] G. Nagy and S. Seth. Hierarchical representation of optically scanned documents. In *Proceedings of International Conference on Pattern Recognition*, volume 1, pages 347–349, 1984.

[14] S. Ray Choudhury and C. L. Giles. An architecture for information extraction from figures in digital libraries. In *Proceedings of the 24th International Conference on World Wide Web Companion*, pages 667–672. International World Wide Web Conferences Steering Committee, 2015.

[15] M. Shao and R. P. Futrelle. Recognition and classification of figures in pdf documents. In *Graphics Recognition. Ten Years Review and Future Perspectives*, pages 231–242. Springer, 2006.

[16] S. N. Srihari. Document image understanding. In *Proceedings of 1986 ACM Fall Joint Computer Conference*, ACM '86, pages 87–96, Los Alamitos, CA, USA, 1986. IEEE Computer Society Press.