Recognizing Figure Labels in Patents

Ming Gong[1,2], Xin Wei[3], Diane Oyen[2], Jian Wu[3], Martin Gryder[3], Liping Yang[4]
[1]University of Dayton
[2]Los Alamos National Laboratory
[3]Old Dominion University
[4]University of New Mexico

Abstract

Scientific documents often contain significant information in figures. The United States Patent and Trademark Office (USPTO) awards thousands of patents each week, with each patent containing on the order of a dozen figures. The information conveyed by these figures typically include a drawing or diagram, a label, caption and reference text within the document. Yet associating the short bits of text to the figure is challenging when labels are embedded within the figure, as they typically are in patents. Using patents as a testbench, this paper highlights an open challenge in analyzing all of the information presented in scientific/technical documents - namely, there is a technological gap in recognizing characters embedded in drawings, which leads to difficulties in processing the text associated with scientific figures. We demonstrate that automatically reading the figure label in patent diagram figures is an open challenge, as we evaluate several state-of-the-art optical character recognition (OCR) methods on recent patents. Because the visual characteristics of drawings/diagrams are quite similar to that of text (high contrast, width of strokes, etc), separating the diagram from the text is challenging and leads to both (a) false detection of characters from pixels that are not text and (b) missed text that is critical for identifying the figure number. We develop a method for automatically reading the patent figure labels by first identifying the bounding box containing the label using a novel non-convex hull approach, and then demonstrate the success of OCR when the text is isolated from the diagram.

Introduction

Recognition of text that is embedded in an image is a well-studied problem, which can be split into two separate problems: text detection — identifying regions of the image that contain text — and optical character recognition (OCR). OCR methods generally assume that majority of the pixels in the image are text and maps groups of pixels to characters. When the image contains non-text elements, it is best to first segment the regions containing text using text detection methods. Typically, text recognition is carried out following the basic pipeline of (1) text region detection which may contain false-positive regions, followed by (2) OCR on each text region and (3) filtering of OCR results to discard regions in which OCR fails to produce reasonable character recognition results. By following this pipeline, OCR is more successful without having distracting non-text pixels present in the image, while false-positive regions found in step (1) can be filtered out by step (3).

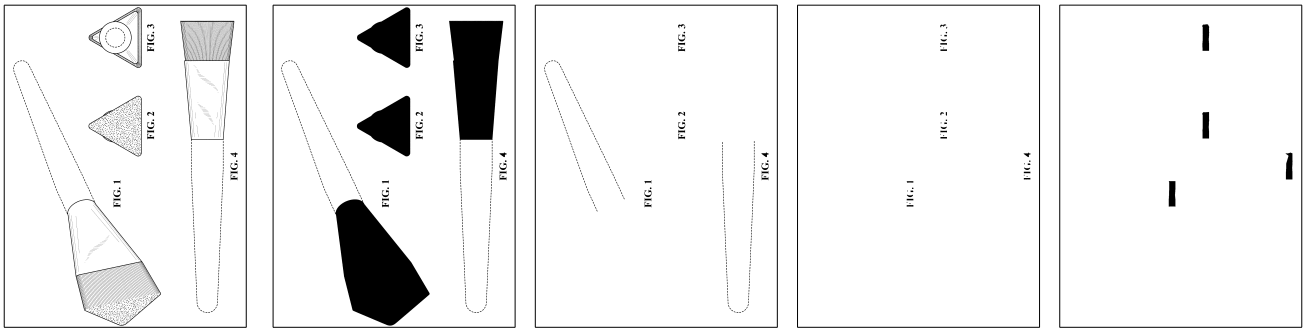
We evaluate open-source and commercial OCR methods on a set of patent drawings with the goal to recognize all figure labels embedded within each patent figure, so that we can associate figure labels with specific drawings. Commercial methods significantly outperform open-source OCR, but even at a recall level of 0.95 there is room for improvement before such methods can be deployed at the scale needed to analyze a large number of patents. We find that open-source OCR works well when text is isolated from the rest of the figure, and therefore we develop a method to isolate figure label text in patent figures.

Label Text Detection using α-Shapes

Patent figures are generally composed of a drawing and label text; or several drawings and multiple label texts (Figure 1). These drawings and labels are spatially located on the page such that it is easy for a person to read the figure label and associate it with the corresponding drawing; yet this remains challenging for computer vision and OCR methods. Our goal is to automatically segment patent figures into regions of drawing and text. We find that existing OCR methods often fail to recognize labels in patent drawings, primarily due to patent drawings being mainly composed of lines, e.g., strokes, curves, which have similar visual characteristics to text, such as high foreground/background contrast, sharpness of edges, and ratio of foreground/background pixels.

In patent figures, the text labels occupy a reasonably compact region of the figure. Therefore, we propose to identify text regions using α-shapes. An α-shape is the smallest polygon that encloses all the points (foreground pixels) in an area, similar to a convex-hull, but with an α allowance for non-convexity (Edelsbrunner, Kirkpatrick, and Seidel 1983). However, α-shape calculation is computationally expensive for a large number of foreground pixels. We therefore develop a workflow that simplifies the figure to candidate regions of text, then removes dashed lines through morphological erosion and then isolates text regions with α-shapes.

Patent figures are processed first by converting to a binary image and then closed regions are filled and filtered out by



(a) Input image binarized (b) Closed regions filling (c) Label candidates (d) Dashed lines removal (e) Generated α-shapes

Figure 1: Identifying label regions in an example patent figure.

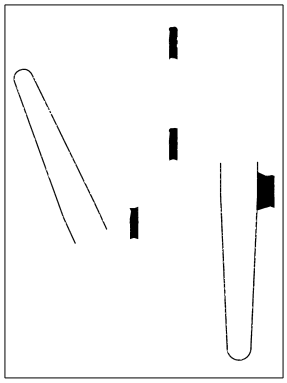


Figure 2: Example of α-shapes without dashed line removal shows text being merged with the nearby dashed line.

size (text contains small enclosed regions inside loops like the number "0", so we filter out large filled regions). However, technical drawings contain not only closed shapes, but also lines, especially dashed lines in many cases. Dashed lines will cause the α-shape to enclose not only the figure label, but also parts of the drawing, as demonstrated in the right-most α-shape of Figure 2. Therefore, dashed lines have to be removed before generating the α-shape to ensure the segmented label regions only include label text. Dashed-line removal is achieved by morphological erosion both horizontally and vertically with a 1 pixel-wide kernel.

With large enclosed regions and dashed lines removed, the α-shapes are generated and employed as a mask applied against the input image for label region candidates segmentation. Each label region candidate is then fed to OCR for text recognition. Here, we use Tesseract (Smith 2007), which is an efficient and portable OCR implementation that is widely used. However, Tesseract is unable to recognize rotated text in small images. Patent figures are often rotated 90 degrees counter-clockwise; therefore, a figure label region candidate with a height greater than its width is automatically rotated 90 degrees clockwise before being fed to OCR.

After text is recognized, a filter determines if the given region candidate is a figure label region or not by checking whether the text contains "Fig" (non case-sensitive). Then the qualified text information is preserved and the coordinates of the corresponding figure label region, including coordinates of original image and the ones of rotated image if any, are also recorded.

As shown in Figure 1, the steps for extracting labels from patent figures are:

1. Segment drawing regions and candidate text regions.
   (a) Threshold image to black on white using Otsu method, Figure 1(a) (Otsu 1979; Sezgin and Sankur 2004).
   (b) Fill regions using mathematical morphology to segment regions of content, Figure 1(b) (Gonzales and Woods 2002).
   (c) Filter connected components by size to produce candidate text regions, Figure 1(c).
2. Generate α-shape.
   (a) Remove dashed lines by morphological erosion both horizontally and vertically, Figure 1(d).
   (b) Generate α-shapes of neighboring pixels to identify candidate bounding polygons of label text, Figure 1(e).
3. Read label.
   (a) Apply OCR to candidate text regions (Smith 2007) (with regions rotated 90-degrees clockwise if the region height is greater than its width).
   (b) Filter regions based on whether the recognized text fits a rule-based pattern of figure labels.

The obtained text mask facilitate the use of label extraction. In addition to facilitating better accuracy of OCR, the text mask allows the removal of text from drawings to facilitate increasing the accuracy of computer vision approaches to visual similarity comparisons by reducing the impact of figure labels.

SWT Method Existing text region detection methods include stroke width transform (SWT) (Epshtein, Ofek, and Wexler 2010) and the deep-learning based "efficient and accurate scene text" (EAST) detector (Zhou et al. 2017). These approaches generally look for regions of high contrast within an image and have been demonstrated to work well for text contained in natural images (such as reading street

signs), as well as for text annotations pasted on other images; where sharp transitions between the texture of text and non-text regions are evident. Of these, SWT is the most similar to α-shape because it does not rely on machine learning; and instead evaluates the width of strokes themselves (i.e. using shape information).

Evaluation of Label Recognition

Ground Truth Corpus

We build the ground truth by annotating figures in US patents, downloaded from patents.reedtech.com. Each patent folder includes an XML file that contains full text marked up with XML tags and a number of TIF figure files. Patents are grouped into different types, such as DESIGN, PLANT, and UTILITY. Random sampling from all types of patents results in a heterogeneous corpus including tables, flow charts, architecture diagrams, etc. In this study, we focus on figures in the DESIGN category because they form a relatively homogeneous sample set consisting of technical drawings, such as the examples shown in Figure 2. As a pilot study, the final ground truth corpus consists of 100 figure files randomly selected from 100 USPTO DESIGN patents approved in January, 2020. We convert the original TIF files to PNG files because all methods accepts PNG as input but not all are compatible with TIF format. A figure file may contain up to four figures with associated labels. Examples are shown in Figure 4. The aspect ratio ranges from 0.122 to 3.555. The size of the PNG files ranges from 89kB to 1.1MB. The size of the TIF files ranges from 21.5kB to 1.3MB.

We created the ground truth corpus by manually inspecting each figure file. The original capitalization (e.g., "FIG." and "Fig.") and number format (e.g., "3.4") were preserved. The final corpus contains 126 figures in 100 figure files.

Methods Compared

We compare eight methods, including four open-source methods (the α-shape based method, the SWT-based method, the EAST-based method, and Tesseract), and four commercial methods (Abbyy FineReader, Adobe Acrobat, Amazon Textract, and Google vision API). To make recognition results solely dependent on the methods but not the input, we use PNG files as input for all methods. The output may contain irrelevant or gibberish characters in addition to figure labels, so we apply regular expressions to parse figure labels. The strings parsed are compared against labels in the ground truth. The precision is calculated as the number of correctly identified labels divided by the total number of labels identified by an OCR method (the parser can accurately find all labels if they exist). The recall is calculated as the number of correctly identified labels divided by the total number of labels found in the ground truth, which is 126.

For the text detection methods (SWT and EAST) which do not in themselves perform OCR, we send the results from text detection to Tesseract for OCR (similar to step 3a in our α-shape method). The output of SWT is an image with only the text pixels present (non-text pixels are filtered out); and so Tesseract receives this one image for OCR. SWT is unable to detect whether text is rotated to the correct orientation, and so we provide hand-rotated images to SWT. The output of EAST is a bounding box for each text region in an image, and so Tesseract receives just the region of the image within this bounding box, rotated so that the bounding box is wide rather than tall (and we concatenate the results for all bounding boxes of an image).

The results in Figure 3 indicate that the Google vision API outperforms the other methods achieving $F_1 = 0.94$, followed by α-shape ($F_1 = 0.91$), Amazon Textract ($F_1 = 0.88$), Adobe Acrobat ($F_1 = 0.83$), and Abbyy FineReader ($F_1 = 0.77$). Although commercial methods seem to achieve outstanding performance, the technology behind them is proprietary and the cost can be high to process a large volume of figures in patents and other types of scholarly documents. The α-based method achieves a better performance than Amazon Textract and Adobe Acrobat. One common characteristics among all methods except SWT is that they all achieve relatively high precision (0.96–1.00) but the the difference of recall values is more diverse (0.44–0.89).

We also compare the performances of methods that support figures in TIF format, including Tesseract, Adobe Acrobat, SWT-based, and α-shape-based methods. Generally, results are similar or worse when using TIF as an input rather than PNG, and therefore we do not show them. This is somewhat surprising as the conversion from TIF to PNG could lose information, but it seems that the extra detail contained in the TIF format is not helpful for OCR.

It is interesting that Google vision API failed to extract labels containing only numbers, e.g., Figure 4(b). It also fails to extract labels of figures in which the labels are rotated by ± 90 degrees (Figure 4(c)). Many of Adobe's failure cases are due to detecting an incorrect rotation of the image - Adobe's OCR method only appears to work on upright text. Abbyy and Tesseract have difficulty extracting labels if they are close to the object, e.g., Figure 4(a), or if the labels are in a box surrounding the object, e.g., Figure 4(d). Most labels extracted using Google vision API are relatively clean with little or no noise characters. Abbyy and Amazon produce a little more noise characters. Tesseract produces a relatively large amount of noisy characters. All imply the importance of segmenting a figure into text and objects if the label recognizer is built on Tesseract.

One advantage of the α-shape-based method is to isolate text and figures, which is why it achieves an outstanding performance. For example, the label in Figure 4(e) is not extracted by Google vision API, but is successfully extracted by the α-shape-based method. An error analysis of this method indicates that most errors are caused by Tesseract. For example in Figure 4(c), Tesseract reads "Fi.", instead of "FIG.3". In some cases, it is hard to determine the kernel size for removing dashed lines because of the existence of both short and long dash lines, e.g., Figure 4(a), in which our method failed to recognize "FIG. 4" because the α-shape enclosed the dash line point above "FIG. 4", which confuses Tesseract. The other text detectors tend to detect the "FIG" text but not the number, and this accounts for the very low

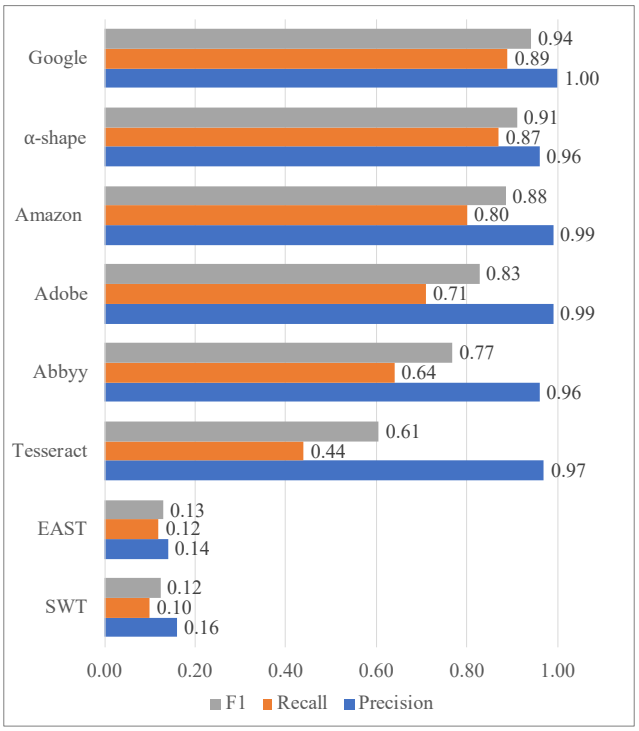


Figure 3: A comparison of precision, recall, and F_1 of different OCR methods on the ground truth corpus.

performance. The output from SWT includes quite a few areas of false-positive text detection and so the OCR output can be quite messy. The EAST detector performs quite well at detecting the "Fig" text no matter the rotation, with only a few false-positive regions, but misses the number.

Discussion

Our results indicate that accurately extracting figure labels is an open question, at least for open-source software. One challenge is to automatically rotate figures so that the label text is in the right orientation before fed to OCR. Another challenge is isolating text from drawings. The α-shape-based method, which beats Google vision API in certain cases, e.g., Figure 4(e), provides a promising solution.

Summary

In this study, we compared eight open-source and commercial OCR methods in figure label extraction as evaluated on a small sample of figures from US DESIGN patents. We also developed a heuristic method based on α-shape. Although commercial methods achieve the highest precision and recall, the open-source α-shape-based method, achieves a comparable performance. We argue that developing a self-adaptable open-source framework for figure label detection is still an open challenge. Future work includes developing learning-based models to adapt kernel size and α parameters to different label characteristics.

The data and source code used in this work are publicly

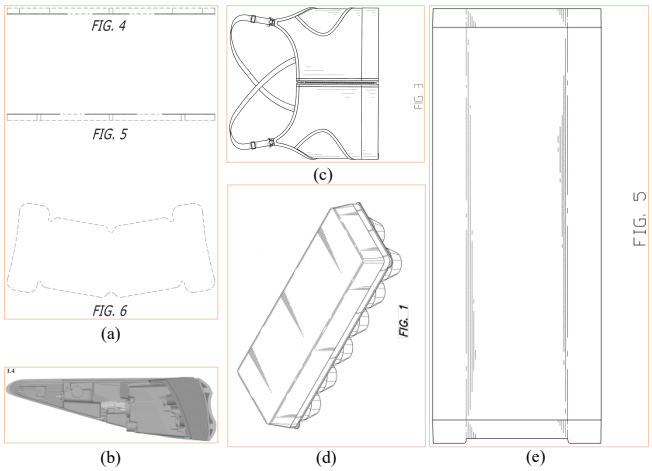


Figure 4: Examples of challenging cases for Amazon Textract, Google vision API, and the α-shape-based methods. An orange border is added to mark figure boundaries.

available on Figshare[1] and GitHub[2].

Acknowledgments

Research conducted by MGong and DO presented in this paper was supported by the Laboratory Directed Research and Development program of Los Alamos National Laboratory under project number LDRD20200041ER. Research conducted by XW, JW, and MGryder presented in this paper was supported by Los Alamos National Laboratory subcontract BA601958 awarded to Old Dominion University.

References

Edelsbrunner, H.; Kirkpatrick, D.; and Seidel, R. 1983. On the shape of a set of points in the plane. *IEEE Transactions on Information Theory* .

Epshtein, B.; Ofek, E.; and Wexler, Y. 2010. Detecting text in natural scenes with stroke width transform. In *IEEE Conference on Computer Vision and Pattern Recognition*.

Gonzales, R. C.; and Woods, R. E. 2002. Digital Image Processing.

Otsu, N. 1979. A threshold selection method from gray-level histograms. *IEEE Transactions on Systems, Man, and Cybernetics* .

Sezgin, M.; and Sankur, B. 2004. Survey over image thresholding techniques and quantitative performance evaluation. *Journal of Electronic Imaging* .

Smith, R. 2007. An Overview of the Tesseract OCR engine. In *IEEE Intl Conference on Document Analysis and Recognition*.

Zhou, X.; Yao, C.; Wen, H.; Wang, Y.; Zhou, S.; He, W.; and Liang, J. 2017. EAST: An Efficient and accurate scene text detector. In *IEEE Conference on Computer Vision and Pattern Recognition*.

[1] https://doi.org/10.6084/m9.figshare.13416311.v1
[2] https://github.com/GoFigure-LANL/patent-label