Bootstrapping Text Recognition from Stop Words

Tin Kam Ho
Bell Laboratories, Lucent Technologies
700 Mountain Avenue, 2C-425, Murray Hill, NJ 07974, USA

Abstract

Recognition of arbitrary noisy English text has been difficult because of problems in character segmentation and multi-font symbol classification. Both segmentation and recognition can be easier with more knowledge of the dominant font used in a given text page. This has led to some recent studies that show promising methods for extracting character prototypes from a text image provided that truth is given for part of the image. In this paper we investigate the feasibility of such a strategy without dependence on ground truth. We replace the needed truth by results of direct recognition of some frequently occurring words. The method makes use of the observation that over half of the words in a typical English text passage are contained in a very small lexicon.

1 Introduction

Training a symbol classifier for a wide range of shape variations and image degradations has been a well-known challenge in machine-print OCR. An alternative strategy is to train the classifier only for the fonts and degradation conditions present in a given text page. Such a strategy is appealing since it can greatly simplify the symbol recognition task, and it is promising for better recognition accuracy. The strategy is practical since in most applications there is usually a dominant font in a given page. It is especially valuable in the cases that multiple pages from the same source need to be processed, as common in many digital library construction tasks.

An adaptive OCR strategy requires some knowledge of the dominant font and defects in a given page image. It is often sufficient to represent such knowledge implicitly by some character prototypes extracted directly from the image. Some recent attempts on adaptive text recognition [12][14] have suggested good uses of such prototypes. In this study, we investigate a multiple pass method that begins by recognizing some common words in a text page, proceeds by using them to extract character prototypes, and continues by propagating accumulated knowledge to recognize the rest of the text. We target the study on lower quality text images like those sent via fax transmission, since for those images the performance is marginal with current state-of-the-art OCR devices.

The method was motivated by the strategy proposed by Nagy and Xu [11][12] in which they assume the input of the truth labels for a small segment of the text to be recognized. Our method does not rely on input of any ground truth. Instead, we make use of an observation that some words occur in almost every page of English text, and we initiate the learning process by first recognizing those words.

Focusing on the common words (which are often referred to as *stop words*) has the advantage of simplicity: there are relatively few of them, so recognition can be based on a very small lexicon, which permits both higher accuracy and speed. On the other hand, it requires a reliable reject criterion so that one knows when a word should not be recognized as one of those in the lexicon. We will detail how this is done through the use of a *decision forest* classifier [5] [6].

Our initial focus on a small set of common words differentiates our method from that proposed by Hong [8], who describes two uses of equivalence of word images: (1) character segmentation by matching equivalent partial words; and (2) OCR error correction by majority vote within word image clusters. In each use there is no direct recognition of word images before character segmentation and recognition, and the algorithm is applied to all words in the page. If all words are considered, direct word recognition can be difficult because of the complexity in building classifiers for tens of thousands of words. By focusing on only the stop words we can afford to employ more expensive holistic word recognition techniques, which we believe are essential for the method to work well on low quality images.

2 Frequency of Occurrence of Stop Words

Although there have been extensive analyses on distributions of English words in linguistics, such statistics are not necessarily useful in OCR. One reason is that statistics derived from large corpora, such as letter, n-gram, or word frequencies, could deviate largely from those observed from small samples, like the relatively short text (one page or a few pages) typically handled by OCR devices. Consequently one has be to very careful in applying probabilistic language models to short text.

However, there is one observation that holds true surprisingly for very short text passages: if one makes a list of the top 135 most common words from modern American English usage, and takes any random word from an American English sentence, there is a greater than 50% chance that this word will be contained in the list [10]. This fact is used in a recently proposed method to decode short cryptograms [2].

To see if the same holds true for conditions encountered by OCR devices, we counted the occurrences of such words using a collection of text samples that are of length typical of those processed by OCR. For consistency with our target on low-quality images, we chose the fax images of the Business Letter Sample of the UNLV database as our test set. Previous OCR results on this database have been reported in [13]. There are 200 pages in this collection. We made a dictionary using the most frequently occurring 135 words from the Brown Corpus, and then counted their occurrences in the ground truth of those images.

Figure 1 (a) and (b) show the per page count and fraction of words that are contained in this dictionary. Except for a few pages containing very short business letters, we found the hypothesis is roughly true: most pages have from 40% to 50% of the words occurring in the dictionary we specified. The slight deviation from the expected 50% is understandable since there are many non-sentences (such as dates and address blocks) in business letters. On average, each page has 248.75 words, and about 43% of them are contained in the dictionary. Even in the shortest letter, 12 of the 25 words are found in the dictionary. Thus a strategy based on recognizing these words will be applicable to every such page.

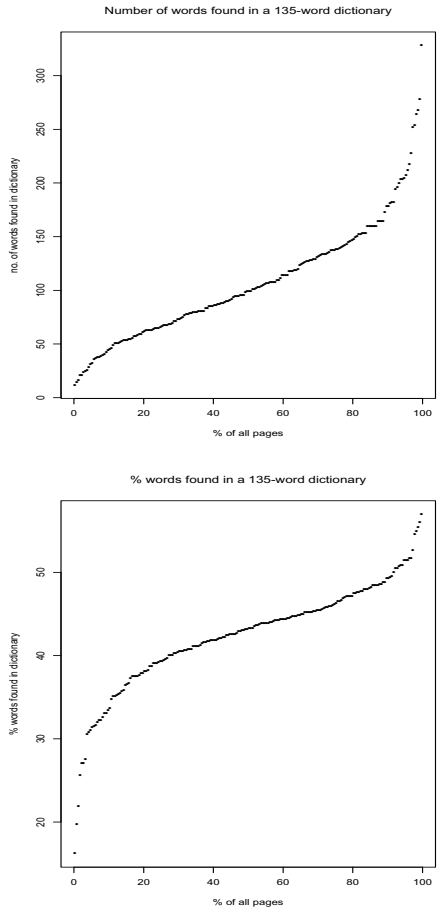


Figure 1: Per page count and fraction of words contained in a 135-word dictionary.

3 Recognition of Stop Words

To initialize the learning process, we first attempt to recognize this small set of 135 common words from a given page image. For convenience we will refer to them as stop words. In order that accuracy will not depend on reliable character segmentation and recognition, which is the challenge for low-quality images in the first place, we use "holistic" word recognition techniques to identify these words. Such techniques treat a word as a single symbol, and recognition is by matching the whole shape of a word image to pre-fabricated prototypes [3] [4].

Four commonly used fonts are used to create the word prototypes: Times-Roman, Courier, Helvetica, and Computer Modern Roman. First, synthetic images of words in the stop word lexicon are composed using font prototypes according to usual typographic rules. Degraded images of the synthetic words are then generated pseudo-randomly using a document image defect model [1]. The procedure is repeated for each word and each font. Altogether, 400 samples are created for each word representing 4 fonts, 4 point sizes (8,10,12,14), and 25 tuples of pseudo-random defect parameters. The output resolution is set at 200x200 to simulate (fine mode) fax quality. To extract image features, each word image is size-normalized to 16 rows $\times$ 64 columns. All white gaps between characters are removed before size normalization. Figure 2 shows some size-normalized synthetic word samples.

The feature vector is the concatenation of four vectors: (1) a "binary subsamples" vector, (2) a "pixel correlation" vector, (3) a "vertical runs count" vector, and (4) a "horizontal runs count" vector.

To compute the subsamples, the original image is scanned from left to right and top to bottom; in the process each 2$\times$2 (nonoverlapping) region is represented by 1 if three or more pixels are black and 0 otherwise. The same process is then repeated for the reduced image until there is only one remaining row or column. The pixel correlation features are conjunctions and disjunctions of neighboring pixels in various directions, and are detailed in [5]. The other two vec-

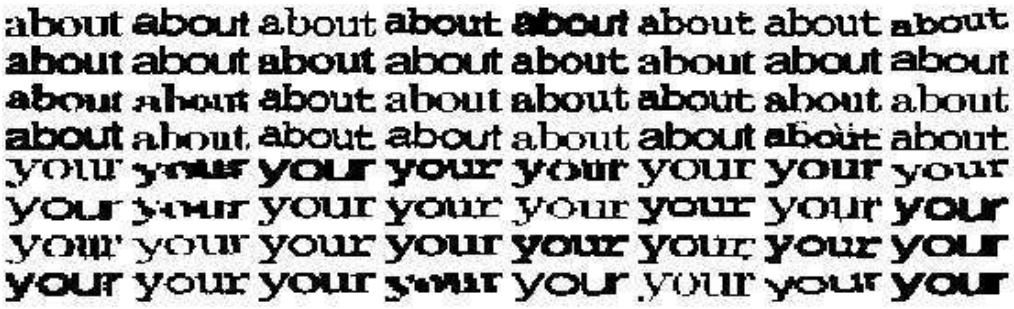


Figure 2: Synthetic samples of two words created using the defect model.

tors are simply counts of black runs (contiguous blocks of black pixels) in each column and row of the normalized image.

The feature vectors describing the synthetic images are then used to construct a decision forest. The choice of a decision forest as the classifier is motivated by its accuracy, speed, and its natural confidence measure which can be thresholded to identify reliable decisions. For better resolution of the confidence measure, similar word samples are created for three more fonts: ITC Bookman Medium Roman, Computer Modern Sans Serif, and Computer Modern Typewriter. Samples of these fonts are used to populate the forest after construction.

In this initial trial, words are synthesized only in lower case, so words in upper case, italics, and those with an initial capital letter are omitted and will be rejected. This is consistent with our focus on the dominant font and our emphasis on classifier simplicity. We expect that this does not cause much difficulty in practice since in the majority of applications most words in a text page are in lower case. We will describe back-up procedures later for recognition of words and characters in non-dominant fonts.

An input image first goes through page layout analysis that extracts the text lines and locates the words. The stop word classifier is applied to each word. To measure the accuracy of the classifier, we applied it to all 200 pages in the faxed business letter sample. For each image, we counted the number of word images that are recognized reliably, i.e., those with decisions passing our confidence threshold. An exception was made that all recognized "i"'s were abandoned, since they are often non-word fragments resulted from noise stretched by size-normalization or incorrect word segmentation.

Among the 200 images, one page has no reliable word decisions (the text is too short). Others have on average 47.6 reliably recognized word images per page. These images represent, again on average, 20.7 distinct words per page. Altogether there are 9469 reliably recognized word images. Only 13 of the decisions are wrong. That translates into a correct rate of 99.86%. It is possible that the correct rate can be further improved by using more image features or in combination with other techniques [3] [4], but the tradeoff between speed and accuracy is an important consideration. Instead, we prefer to pick up the errors in a later stage when the words are used to extract character prototypes.

4 Extraction of Character Prototypes

Once some stop words are recognized and determined to be reliable, we apply the prototype extraction procedures similar to those proposed in [11] and [12] to those words. This phrase of learning consists of four steps: (1) character width estimation, (2) word shifting and matching, (3) common character extraction, and (4) bitmap averaging.

The widths of the recognized word images (those passing the confidence threshold) and the characters in the decided (word) classes are first used to estimate the mean width of each included character. This is done by forming a system of linear equations, where the variables are the widths of each character that occurs in the decided words, plus one variable for the inter-character space that is assumed to be the same between all pairs of neighboring characters. The constant spacing assumption is a rough approximation, and a refinement of the model is possible using the methods proposed in [9]. The least squares solution for the linear system is obtained, which gives the width estimates. These estimates are important hints for character segmentation and extraction, especially for proportional fonts.

We then select pairs of recognized words that share one and only one common character. Guided by the estimated character widths and inter-character spacing, we determine regions of the words that are expected to contain the target character. The two regions are then shifted and matched column by column. For any two columns (one from each word), the fraction of pixels that are black in both columns over the total number of black pixels is used as a score of similarity. The columns are aligned by the estimated



Figure 3: Two pairs of words aligned for extraction of common characters ("a" in the left pair, "n" in the right pair).

baseline of the both words, and a small local search is performed for the best vertical shift. The column-wise scores are summed over the width of the region which is dependent on the estimated character width. The horizontal shift that produces the minimum of these sums is used to align the two words. The region over which the column-wise score is above a threshold is extracted as the character prototype. Figure 3 shows two pairs of words aligned for extraction of a common character. Further refinement of this process is possible using estimates of position probabilities proposed in [11][12].

In many cases more than one prototype are extracted for the same letter. For example, this occurs for every letter in Figure 3. The bitmaps for the same letter can be averaged to remove some random noise. Bitmap averaging can be as simple as just aligning by centroids followed by majority voting, or as sophisticated as we attempted in a previous experiment [7] where alignment is performed in sub-pixel accuracy.

Using the shift and match procedure on the reliably recognized stop words from our test set, on average, prototypes of 9.9 distinct character classes per page are extracted. This has made use of a reject criterion that discards pairs of words that do not produce a high enough similarity score even at the best alignment. The reject criterion serves as a safeguard against misrecognized words or pairing of words in different fonts.

Prototypes obtained from the pairwise matching of words can be used to initiate a recursive segmentation process that aims at completely segmenting the recognized stop words. In this process, word images are first segmented at the position where a character has been extracted, with the decision string divided accordingly. Then characters obtained from other words are shifted on top of the partial word images for further matching and segmentation. The process continues until no prototypes are available to match the remaining partial images [8].

5 Using Character Prototypes in Text Recognition

The character prototypes extracted from the stop words can be used in a variety of ways. They can be used to train or adapt a classifier on-the-fly, or matched to a font database to estimate the dominant font in the page. If sufficient prototypes are obtained, they can be used to synthesize word shapes for a larger lexicon for recognition of the remaining words. It is also possible to use the prototypes directly in a dynamic programming approach, such as the level-building algorithm attempted in [12]. In any case, the recognition problem is reduced to a quasi-single-font problem, which opens ways to a wide range of solutions and has potential for much better accuracy.

6 The Effect of Missing Characters

Although elaborate procedures can be applied to extract characters from the recognized stop words, there is always a chance that some characters will still be missing because they are not contained in the recognized words. There are several ways to proceed: (1) use a default multi-font classifier on the unknown characters; (2) estimate the font and retrieve the missing characters from a matched font sample; and (3) try to resolve the ambiguity by context. In method (1), the multi-font classifier needs to be accurate enough to resolve any ambiguity left after using word context. Font estimation in method (2) is difficult, but we can aim at only classifying the character prototypes into some broad font families.

To determine the extent that word level context can help, we experimented with the ground truth of our test pages. Assuming that all stop words are recognized and all characters are extracted from them, we replaced all missing characters in the rest of the words with a symbol denoting an unknown character. Then we performed regular expression matching against a lexicon of 43,300 words (list of all words from the Brown Corpus).

On average, 5.92 characters are missing from all those included in the stop words, and as a result, 42 of the 248.75 words (or 16.8%) on a typical page contain unknown characters. Most of those words can be uniquely matched to words of the same length in the lexicon. Some others have multiple matches of the same length, and for them the average size of the retrieved neighborhood is 7.21 words. To decide among the words in the neighborhood, multi-font techniques needed to be involved. On average, only 4.13 words per page have no matches of the same word length. Those are mostly words with inflections (such as "technologies"), proper nouns, abbreviations (such as "RSVP", "PhD"), or newer words (such as "programmable", "debugger")[1]. These words need to be matched by approximate string matching or on a character-by-character basis using a multi-font classifier. The results suggest that, with the help of a general purpose lexicon and without reliable font estimation, only 15.4 (11.27+4.13) or 6.19% of the words on a typical text page need to be recognized by multi-

[1]Numeral and other non-alpha strings have been excluded in this study.

font techniques. Table 1 details the results of matching the words containing missing characters to the 43,300 word lexicon.

Table 1: Results of regular expression matching with a 43,300 word lexicon (averaged over 200 pages).

No. of words that		
are present on a test page	248.75	100.00%
have missing characters	42.04	16.90%
have a unique match of same length	26.74	10.75%
have multiple matches of same length	11.27	4.53%
have no matches of same length	4.13	1.66%

7 Conclusions

We propose an adaptive text recognition strategy that is initiated by recognizing some common (stop) words. The recognized words are then aligned for extraction of common characters, and then recursively segmented using accumulated character prototypes. Character prototypes obtained from processing the stop words can then be used to train a classifier to recognize the rest of the text. By focusing on a relatively small set of words initially, more expensive word recognition techniques such as holistic word shape matching can be applied, so that the method can be applied to lower quality images. In several stages reject criteria are introduced to defend against unreliable recognition and character or word matches.

The strategy uses a characteristic mix of whole word and isolated character based methods. Also, other techniques such as string matching and dynamic programming are found useful in the process. Moreover, we do not intend to completely abandon multi-font classifiers. Instead, we consider them as a supplementary tool for ambiguities that cannot be solved by the proposed procedure. Those include text in a non-dominant font (such as italics, boldfaces, numbers, or headers) and words with missing characters that cannot be matched uniquely to words in a general purpose lexicon. The study of the effect of missing characters shows that the use of a multi-font classifier can be reduced to a minimum.

Acknowledgements

The author thanks George Nagy for reference [2], and him and YiHong Xu for helpful discussions and sharing of their prior work; Henry Baird for the defect model; John Hobby for trials of his bitmap matching and averaging algorithms; and Dan Lopresti and Raffaele Giancarlo for discussions on string matching.

References

[1] H.S. Baird, "Document Image Defect Models", in H.S. Baird, H. Bunke, K. Yamamoto (Eds.), *Structured Document Image Analysis*, Springer–Verlag, pp. 546-556, 1992.

[2] G. Hart, To Decode Short Cryptograms, *Communications of the ACM*, **37**, 9, September 1994, 102-108.

[3] T.K. Ho, *A Theory of Multiple Classifier Systems and Its Application to Visual Word Recognition*, Doctoral Dissertation, Dept. of Computer Science, SUNY at Buffalo, 1992.

[4] T.K. Ho, J.J. Hull, S.N. Srihari, A Computational Model for Recognition of Multifont Word Images, *Machine Vision and Applications*, **5**, 1992, 157-168.

[5] T.K. Ho, Random Decision Forests, *Proceedings of the 3rd International Conference on Document Analysis and Recognition*, Montreal, Canada, August 14-18, 1995, 278-282.

[6] T.K. Ho, C4.5 Decision Forests, *Proceedings of the 14th International Conference on Pattern Recognition, Brisbane*, Australia, August 17-20, 1998.

[7] J. Hobby, T.K. Ho, Enhancing Degraded Document Images via Bitmap Clustering and Averaging, *Proceedings of the 4th International Conference on Document Analysis and Recognition*, Ulm, Germany, August 18-20, 1997, 394-400.

[8] T. Hong, *Degraded Text Recognition Using Visual and Linguistic Context*, Doctoral Dissertation, Dept. of Computer Science, SUNY at Buffalo, 1995.

[9] G.E. Kopec, Least-Squares Font Metric Estimation from Images, *IEEE Transactions on Image Processing*, **2**, 4, October 1993, 510-519.

[10] H. Kucera, W.N. Francis, *Computational Analysis of Present-Day American English*, Brown University Press, Providence, R.I., 1967.

[11] G. Nagy, Y. Xu, Priming the Recognizer, *Proceedings of the IAPR Workshop on Document Analysis Systems*, Malvern, PA, October 14-16, 1996, 263-281.

[12] G. Nagy, Y. Xu, Automatic Prototype Extraction for Adaptive OCR, *Proceedings of the Fourth International Conference on Document Analysis and Recognition*, Ulm, Germany, August 18-20 1997, 278-282.

[13] S.V. Rice, F.R. Jenkins, T.A. Nartker, The Fifth Annual Test of OCR Accuracy, Technical Report 96-01, Information Science Research Institute, University of Nevada, Las Vegas, April 1996.

[14] A.L. Spitz, An OCR Based on Character Shape Codes and Lexical Information, *Proceedings of the 3rd International Conference on Document Analysis and Recognition*, Montreal, Canada, August 14-18, 1995, 723-728.