TEACHING A COMPUTER TO READ

George Nagy
Rensselaer Polytechnic Institute
Troy, NY 12180, USA

Abstract

Current OCR devices generate too many errors due to missegmentation and multifont confusion. Some of these errors can be avoided or corrected though greater use of linguistic context and local shape consistency in printed matter. To teach machines to read, OCR designers should allow their devices to modify their internal parameters according to information gained through a family of internal feedback loops.

1 Introduction

It is not difficult to design an optical character recognition (OCR) system that will recognize, almost perfectly, well-formed, well-spaced, and well-scanned printed characters. However, even the best of the commercial devices barely reach 99 per cent correct recognition when faced with poorly-printed or poorly-copied dense text in any of the commonly used sizes and typefaces [4]. One error per hundred means an average of one unrecognized character in every other line of a column of text! With such a high error rate, the cost of finding and correcting errors and rejects is prohibitive, and it is less expensive to key-in the entire column than to resort to OCR.

Feature extraction and classification methods for isolated characters have been the objective of intensive study for upwards of thirty years, so we we have little to hope from that direction. What aspect of printed text can we then exploit to achieve more accurate OCR?

Two aspects of printed matter do not appear to have been exploited yet to full extent. One is the use of linguistic context, interpreted in the broadest sense to include all the conventions of written communication. The second is spatial context: the shapes of the letters exhibit local consistency in each passage of text because typefaces and typesizes are not changed arbitrarily, and because each page-image is generated by the same printing, reprographic, and scanning device. However, to take advantage of either of these observations, OCR devices must have a broader window for classification than a single character.

For high-quality printed matter, we have been able to show that clustering the pattern shapes by virtue of their spatial consistency, and then assigning a label to each cluster using linguistic context, allows correct interpretation without *any* prior knowledge of the shapes of the characters [1, 2, 3]. These schemes, which essentially transform the OCR problem into a substitution cipher problem, need to be modified and extended greatly for application in a realistic OCR environment.

After a brief description of the most common OCR errors, we will discuss increased use of linguistic context, spatial context, and the adaptation of a classifier to a single typeface in a multifont environment. All of these ideas aim at more wholistic recognition. We shall then propose a systematic approach to a simple form of machine "learning" which may, perhaps, serve as a paradigm for other pattern recognition systems as well.

2 Common OCR errors

A large fraction (30-50 per cent) of the errors made by current OCR systems on printed technical documents cannot be characterized as confusions between two characters, such as *2* and *Z* [5]. Because they are the result of missegmentation, there is no one-to-one correspondence between "true" and "recognized" character. Many of the errors are caused by characters fragmented because of light print, gaps introduced by a poorly adjusted copier, insufficient scanning resolution for hairline strokes, or incorrect threshold setting. The fragments introduce superfluous characters in the output of the OCR system: the prototypical example is an *m* interpreted as *rn*.

Commercial devices seem to be able to handle touching characters better than fragments, but omission of a character still accounts for about one third

0-8186-2915-0/92 $3.00 © 1992 IEEE

of all segmentation-related errors. For instance, *rn* might be interpreted as *m*. Many of the segmentation errors appear less plausible to the human reader, although humans are not immune either from errors in interpreting low-quality characters without context. In fact, OCR devices come close to, or even exceed, human performance on isolated characters.

Also frequent are omitted and extra blanks and end-of-line marks. The latter error may be caused by subscripts or superscripts, or by skewed lines. These errors hamstring lexical correction methods based on known word boundaries, decrease the legibility of the text, and may adversely affect automatic information retrieval.

Because of the small size of punctuation marks, errors involving commas, periods, hyphens and apostrophes are prevalent. They may be either missed altogether, or confused with one another.

Characters that do not have an ASCII representation, such as subscripts and superscripts, mathematical symbols, the Greek and Gothic alphabets, and other specialized glyphs, are not treated consistently by commercial OCR devices. Some can be trained for additional symbols.

Altogether, erroneously inserted characters, missed characters, blanks, punctuation errors, and "exotic" characters account for the majority of all errors on printed technical material. Such errors are often ignored in academic research.

The other confusions are what one would expect. The leading culprit is the *I-l-i-1* group, followed by *0-O*, *c-e*, and *a-s-o-r-n*. Because of their high frequency of occurrence in English text the above confusions account for about one third of the misclassifications that do not involve segmentation errors, although there are some other difficult pairs, such as *Z-2* and *S-5*.

Intelligent reading requires more than affixing the correct label to each character in a document. However, the determination of the correct reading order and of the role of the various layout and "logical" entities is the domain of *document analysis*, of which OCR is only one part. Because the focus of this paper is OCR, errors due to improper *zoning* (the delineation of the textual material to be converted to computer-readable text on each page), are not considered.

3 Linguistic context

We shall discuss contextual aids to text conversion in order of increasing scope: morphological, lexical, syntactic, semantic, and pragmatic methods. To gauge the amount of information residing in context, picture an English-speaking typist attempting to transcribe a letter written by hand in Polish.

The *a priori* probability of character classes enters in the classical formulation of Bayesian character classification [6]. For instance, given that there is an equal amount of evidence for the letter *e* and the letter *c*, choose *e*, which occurs more frequently in English.

One of the first to take advantage of the highly non-uniform letter n-gram frequencies was Raviv. For each character in a word, he computed the *a priori* probability distribution by combining the *a posteriori* probabilities of the preceding letters based on shape with bi-gram or tri-gram frequencies [7]. The probability estimates were based on a Markovian model of the language. Later, Toussaint and Shinghal modified the Viterbi algorithm to find the most likely candidate for an entire word, given the classifier-provided probabilities for each character [8]. For a survey of the early work, see [9] and [10]. With the advent of larger computer memories, lexical techniques have largely supplanted morphological methods, but joint symbol probabilities are still useful for recognizing punctuation.

Lexical methods come in two flavors. The more common is post-processing. Here the output of the classifier is checked against a list of acceptable words. Mismatches are either flagged for human correction, or replaced by the "closest" candidate in the dictionary. Ideally, the correction process takes into account the probabilities of various types of errors. This may be accomplished either by accepting a dictionary word only if it consists of high-probability candidates generated by the classifier, or by basing the measure of closeness on known confusion probabilities.

For example, if a word is recognized as "ehase", which does not appear in the lexicon, then it may simply be flagged for operator correction. However, if according to the classifier *c* is the second most probable candidate for the first letter, then it would be safe to accept "chase" from the lexicon. Alternatively, if only the most probable candidate for each character is generated, then "chase" rather than "phase" would be selected from the lexicon because *c-e* confusions are more common than *p-e* confusions.

Less commonly, the lexicon is used during the classification process itself. Partially recognized words are matched against the dictionary to find plausible candidates for the remaining characters. The classification process can then be restricted to the hypotheses generated from the lexicon. The reduction in the size of the search space is particularly effective in the case of

missegmented characters, where sequences of several characters need to be considered simultaneously [12].

The major shortcoming of lexical techniques is that they help least when accuracy is most important, i.e., on proper names, telephone numbers, dollar amounts, or part numbers. Of course, specialized lexicons may include name and street directories, chemical compounds, geographic names, and even part numbers.

Syntactic techniques require consideration of several words at once. Adequate parsers incorporating the most important grammatical constructs are now available. Some were developed for near-English query languages, while others were constructed for automatic translation or speech recognition. Markov chains on words can also provide an approximation to English grammar [14]. Additional syntactic rules govern punctuation, abbreviations, written-out numbers, numerics, chemical formulae, music notation, and other symbol-based means of communication.

The application of simple grammatical rules may readily determine, for instance, that "a case of beer" rather than "a ease of beer" is correct, or that one wishes "to ease into the task" rather than "to case into the task". Highly inflected languages may facilitate grammatical analysis: an excellent example of the application of Italian syntax to OCR on a poorly-printed legal gazette is given in [13].

Syntactic rules cannot differentiate between "casing the tension" and "easing the tension," because both are syntactically equivalent gerunds. However, even minimal semantic knowledge (obtained from a dictionary rather than a lexicon) may suffice to disambiguate the two phrases. Knowledge-representation tools for natural language have been studied for many years in artificial intelligence.

The required knowledge-base may have to be domain specific. Without knowing whether we are in the kitchen or the garage, we may not be able to choose between "the joints are burned" and the "points are burned". In some instance, even dynamic (pragmatic) knowledge may be required: if the fire was flickering, then she was probably out of "wood"; if, on the other hand, the booties were only half finished, then she was out of "wool".

It seems clear that increasingly higher levels of linguistic understanding will have to be applied to reading low-quality printed matter and, *a forteriori*, to hand-printed and cursive writing. In this matter speech recognition is far ahead of OCR.

4 Spatial context

Omnifont OCR refers to the ability of an OCR system to recognize text printed in a large variety of typefaces. However, in most printed documents, each successive character is not selected at random from one of, say, 300 typefaces: uniformity of appearance is the key to good typesetting practice (and also to legible and pleasing handwriting). The typeface, typesize, and shape deformations due to printing, copying and scanning tend to remain the same for long sequences of symbols. Nevertheless, most "omnifont" OCR devices and published algorithms do not take advantage of spatial consistency, and would do neither better nor worse if the typefaces and character deformations were distributed entirely randomly.

For an egregious but plausible illustration of the power of spatial context, consider two typefaces, where all the letters have different shapes except that the letter *I* in typeface A is identical to the letter *l* in typeface B. Such is actually the case in some sans-serif typefaces, where these letters consist of an unadorned vertical bar. It is clear that, if the font is not known ahead of time, then a classifier based *only* on the shape of each letter cannot possibly recognize accurately material printed in either typeface A or B. Assume, further, that no linguistic context is available: we are reading part numbers such as *Im43an, Cn57lm, br58Mt,* etc. However, we can be sure that each part number is printed in a single typeface.

Rather then constructing a classifier to recognize each individual character, let us design a classifier to recognize each pair of letters. Consider now the symbol pairs *Im* and *lm*. These symbol pairs look different in typeface A than in typeface B, because although the I in A is the same as the *l* in B, the *m*'s are different in each typeface. We can therefore construct, at least in principle, a classifier that will recognize the part numbers perfectly, *without* prior typeface information.

Of course, it is not actually necessary to use a classifier based on n^2 classes, where n is the number of symbols in the alphabet. It is sufficient to apply the pair-recognizer whenever an ambiguous character is encountered.

Another application of spatial context in combination with linguistic context is *reject recovery*. It may happen that a particular member of the alphabet is consistently difficult to recognize, either because of its unusual shape, or because of a printer defect. For example, assume that *h* is such a character in the following sentence:

Ahab blew the whistle.

The *h* in *Ahab* cannot be recognized using linguistic context, because "Ahab" is not in the dictionary. It is ambiguous even in "the", since "tie", "the" and "toe" are all correct English words. However, the second letter in "whistle" must be an *h*. Therefore, by comparing the pixel configuration or feature vector of the two ambiguous letters to that of the *h* in whistle, we can identify all the misshapen patterns.

A simpler example is the occurrence of mutilated *c*'s in the word "chuck". If the *c*'s are not recognized, i.e., the classifier reports "_hu_k", then the lexicon will yield "chuck", "chunk", "thunk" and "shuck". However, if the classifier determines that the two unknown letters are the same, then "chuck" is the only acceptable candidate.

To see the role of *adaptation* in spatial context, let us consider two typefaces where the *I* of typeface A is similar, but not identical, to the *l* of typeface B, and the *l* of typeface A is similar to the *I* of typeface B. The *I* of A is, however, quite different from the *l* of A, and the *I* of B is quite different from the *l* of B. We may therefore be able to design a two-font classifier that discriminates perfectly these letters in the training sample, although slight defects will lead to errors in actual operation.

It should be possible to avoid these errors altogether. On any particular document in either typeface, we ought not have any difficulty, since there is no similarity between the *I*'s and the *l*'s. The defects might lead us to confuse I_A with l_B, or I_B with l_A, but these are not choices that can confront us on any single document.

It is indeed possible to avoid these errors by taking advantage of the fact that the majority of the characters of both classes are correctly classified. For instance, one may compare each sample labeled *I* or *l* to all the samples labeled *I* or *l* in the entire document, and assign to it the label of the majority of the most similar samples. Alternatively, one may adapt the two-font classifier to its effectively single-font environment [11]. We may consider this either a form of *bootstrapping*, or of *implicit font recognition*.

Tuning a classifier to its temporarily single-font environment may also enhance throughput. For example, if a multifont classifier operates by comparing each new pattern to many reference patterns of each class, then references that have not been used for a while can be safely deleted. When low recognition scores indicate that a font change has occurred, all the references can be restored and used again until it becomes clear which ones are most appropriate for the current font. A step in this direction has already been taken by commercial devices that keep track of the most recently used reference patterns.

On-the-job training for OCR devices can also be considered in a more global sense. In almost all applications, the output of an OCR device is proof-read and corrected by an operator before it is used. The similarity between the documents processed day after day in a particular OCR application is likely to be far greater than the similarity between these documents and the ones on which the machine was trained at the factory. After every day's operation, there exists a large database of labeled character images that are by definition typical of the application. However, we are not yet aware of any OCR device that looks back on the mistakes it makes each day, and ows to do better on the following day.

5 Conclusion

It has taken each of us several years to learn to read. It is unlikely that we actually remember all the thousands of shapes that each letter can assume, and if we rely on a consistent theory of letter shapes, we cannot formulate it clearly. But we do certainly learn to take full advantage of our knowledge of common word-fragments, words, grammar and meaning. We subconsciously reject interpretations that do not make sense. When deciphering really poor quality material, such as a scrawled postcard, we compare each puzzling fragment to similar shapes where context facilitates interpretation. For rapid and accurate reading, we depend on typesetting and writing conventions that have evolved through the centuries for just that purpose, and on our knowledge of the language.

Teaching these skills to a machine may be even harder than teaching them to an illiterate adult. But perhaps we should go about it the same way. Show the machine a few examples, teach it a little grammar, and let it read, read, read. We should let it compare its output to that of other reading machines. When it makes the same mistake again and again, we should call attention to it and let it correct itself. If this gets too onerous, we should let whatever application the OCR output is used for provide the feedback: the language translation program, the automated filing system, the information retrieval software, or the accounting routines. Any mistakes that they don't catch probably won't matter.

The crucial step to making all this work is to remove the human from the parameter fine-tuning process. Instead, provide a family of feedback mechanisms that

operate on increasingly larger segments of the document and invoke them as necessary. Let the feature extraction process adjust the grey-scale thresholding mechanism to provide strokes of the expected width. Let the classifier adjust the feature extractor until the features make sense. Allow spatial context to modify the classification algorithm to obtain consistent (i.e., single-font) local classification. Let linguistic context override and adjust the resulting single-font classification when appropriate. And when the output does not make sense in the context of the specific application, change the linguistic rules.

We have given limited examples of some of these notions, but a recognition system that incorporates them should not, of course, be programmed on a case-by-case basis. What we need is an elegant and practical theory of classification that takes into account spatial, linguistic, and pragmatic context, much as the Markov model accounts for n-gram letter frequencies. Easier said than done.

Acknowledgment

During the preparation of this paper, the author was a guest of the Information Science Research Institute of the University of Nevada, Las Vegas. He is grateful for the help of Professors J. Kanai and T. Nartker, and Chief Software Engineer S. Rice.

References

[1] R.G. Casey and G. Nagy "Advances in Pattern Recognition" *Scientific American*, Vol. 224(4) pp. 56-71, 1971.

[2] R.G. Casey "Text OCR by solving a cryptogram" *Proc. Eighth Int'l Conf. on Pattern Recognition*, pp. 349-351, 1986.

[3] G. Nagy, S. Seth, K. Einspahr "Decoding substitution ciphers by means of word matching with application to OCR" *IEEE Trans. Pattern Analysis and Machine Intelligence* Vol. 9(5), pp. 710-715, 1987.

[4] S.V. Rice, J. Kanai, T.A. Nartker "A report on the accuracy of OCR devices" *Information Science Research Institute Technical Report, University of Nevada, Las Vegas*, 1992.

[5] T.A. Nartker, J. Kanai, S.V. Rice "A preliminary report on OCR problems in LSS document conversion" *Nuclear Waste Management Conference, Las Vegas*, April 1992.

[6] C.K. Chow "An optimum character recognition system using decision functions" *IRE Trans. Electronic Computers* pp. 247-254, 1957.

[7] J. Raviv "Decision making in Markov chains applied to the problem of pattern recognition" *IEEE Trans. Information Theory*, Vol. IT-3(4), pp. 536-551, 1967.

[8] R. Shinghal and G.T. Toussaint "Experiments in text recognition with the modified Viterbi algorithm" *IEEE Trans. Pattern Analysis and Machine Intelligence* Vol. 1(2), pp. 184-193, 1979.

[9] G.T. Toussaint "The use of context in pattern recognition" *Pattern Recognition*, Vol. 10, pp. 189-204, 1978.

[10] S.N. Srihari "Computer text recognition and error correction" *IEEE Computer Society Press*, 1985.

[11] G. Nagy and G.L. Shelton "Self-corrective character recognition system" *IEEE Trans. Information Theory* Vol. IT-12(2), pp. 215-222, 1966.

[12] C. Paoli and M.T. Pareschi "A system for the automatic reading of printed documents" *Office Information Systems: The Design Process*, B. Pernici and A.A. Verrijn-Stuart, editors, Elsevier, pp. 311-322, 1989.

[13] G. Boccignone, L.Freina, S. Mogliotti, M.R. Spada "Towards an evaluation of an experimental OCR system by means of a complex document" *Proc. Fifth Int'l Conf. on Image Analysis and Processing*, Positano, World Scientific Publishers, pp. 543-550, 1989.

[14] J.J. Hull "Incorporation of a Markov model of language syntax in a text recognition algorithm" *Proc. Symp. on Document Analysis and Information Retrieval*, ISRI-UNLV, Las Vegas, pp.174-185, 1992.