Document Image Layout Comparison and Classification

Jianying Hu Ramanujan Kashi Gordon Wilfong
Lucent Technologies Bell Labs
700 Mountain Avenue, Murray Hill, NJ 07974-0636, USA
{jianhu | ramanuja | gtw}@research.bell-labs.com

Abstract

This paper describes features and methods for document image comparison and classification at the spatial layout level. The methods are useful for visual similarity based document retrieval as well as fast algorithms for initial document type classification without OCR. A novel feature set called interval encoding is introduced to capture elements of spatial layout. This feature set encodes region layout information in fixed-length vectors which can be used for fast page layout comparison. The paper describes experiments and results to rank-order a set of document pages in terms of their layout similarity to a test document. We also demonstrate the usefulness of the features derived from interval encoding in a hidden Markov model based page layout classification system that is trainable and extendible.

1 Introduction

Capturing visual similarity between different document images is a crucial step in many document retrieval tasks, including visual similarity based retrieval, categorization, and information extraction. In this paper we study features and algorithms for document image comparison and classification at the spatial layout level. These features and algorithms can be used directly to answer queries based on layout similarities [2], or as pre-processing steps for some of the more detailed page matching algorithms [5, 7]. Furthermore, they can serve as the first step in document type identification (business letters, scientific papers, etc), which is useful for document database management, document routing and information extraction.

As shown by the examples in Figure 1, page images from different types of printed documents often have fairly distinct spatial layout styles. Furthermore, these layout characteristics are clearly identifiable at very low resolution. This makes it possible to develop fast algorithms for initial document type classification based on block segmentation only, which can then be verified by more elaborate methods using more complicated geometric and syntactic models for each particular class (e.g., [10]).

Measuring spatial layout similarity is in general difficult, as it requires characterization of similar shapes while allowing for variations originating both from design and from imprecision in the low level segmentation process. Zhu and Syeda-Mahmood viewed document layout similarity as a special case of regional layout similarity of general images and proposed a region topology-based shape formalism called constrained affine shape model [11]. As in many other pattern recognition problems, there is a trade-off between using high-level and low-level representations in layout comparison. A high-level representation such as the one used in [11] provides a better characterization of the image, but is less resilient to low-level processing errors, and generally requires a more complicated distance computation. On the other hand, a low level representation, such as a bitmap, is very robust and easy to compute, but does not capture the inherent structures in an image.

In this paper, we propose a novel spatial layout representation called *interval encoding*, that can be viewed as an intermediate level of representation. It encodes region layout information in fixed-length vectors, thus capturing structural characteristics of the image while maintaining flexibility. These fixed-length vectors can be compared to each other through a simple distance computation (the Manhattan distance), and thus can be used as features for fast page layout comparison. Furthermore, they can be easily clustered and used in a trainable statistical classifier for document type identification.

2 Page Layout Comparison

As input to our algorithm, we assume that the document page has been deskewed and segmented into rectangular blocks of text [1], as shown in Figure 1. The block segmented image is partitioned into an m by n grid and we refer to each cell of the resulting grid as a *bin*. A bin is defined to be a *text bin* if at least half of its area overlaps a single text block and otherwise it is a *white space bin*. We let r_i

denote the i^{th} row and index the bins in r_i as $1, \ldots, n$ from left to right. In comparing two page layouts we use a 2-level procedure. The first level consists of a method for computing a distance between one row on one page with another row on the other page. This level will be discussed in detail later. The second level involves finding a correspondence between rows of the two pages that attempts to minimize the sum of distances between corresponding rows. This is accomplished using a dynamic programming algorithm [9], which finds a sub-optimal path by minimizing the total distance in aligning the rows on two pages within a constrained range of vertical shift.

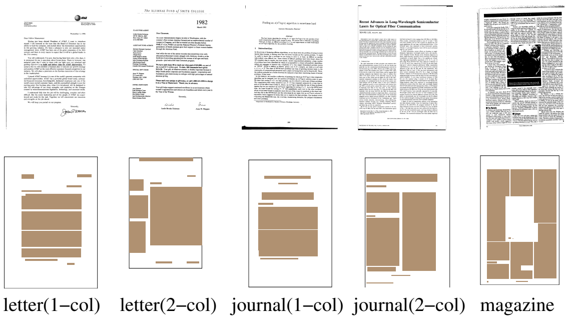

Figure 1. Different types of documents and their block segmentation.

The distance between two rows can be computed in numerous ways. We investigate three methods.

The most natural method is based on the following representation of a row of bins. We define a *block* in a row to be a maximal consecutive sequence of text bins. Suppose the i^{th} row, r_i, has k_i different blocks say, $B^i_1, \ldots, B^i_{k_i}$, ordered from left to right. The bins within a particular block B^i_j are consecutive and we let s^i_j and f^i_j denote the index of the first (leftmost) and last (rightmost) bins in B^i_j respectively. We can represent r_i as a sequence of the pairs $R_i = (s^i_1, f^i_1), \ldots, (s^i_{k_i}, f^i_{k_i})$ and we call such a representation a *block sequence*. Then the distance $d^E(R_i, R_j)$ between two such block sequences R_i and R_j is defined to be the edit distance (see [6]) between them where the cost of inserting or deleting a pair (s, f) is just $f - s + 1$ (the width of the block), and the cost of substituting (s, f) with (u, v) is taken to be $|s - u| + |f - v|$. This distance measure will be referred to as the *edit distance*. While the edit distance appears to be the measure that most accurately captures the differences between rows, it can be computationally unattractive if the lengths of the R_i's become too large. It also has a disadvantage when it comes to clustering. In particular, given a collection of block sequences that form a cluster it is desirable to be able to compute a cluster center, that is, determine a block sequence R that minimizes the sum of the distances from R to each of the block sequences in the cluster. However it is not at all obvious how such a cluster center should be computed. This leaves the unsatisfactory option of choosing as the cluster center one of the block sequences in the given collection that minimizes the sum of the distances to the others.

We introduce a computationally simpler method that overcomes the clustering difficulty of the edit distance. The scheme is based on the Manhattan distance; that is, the l_1 distance where for two vectors $x = (x_1, \ldots, x_n)$ and $y = (y_1, \ldots, y_n)$ the Manhattan distance between them is given by $l_1(x, y) = \sum_{i=1}^{n} |x_i - y_i|$. In order to use the Manhattan distance we need to represent each row as a fixed length vector. In fact, each row, r_i, will be represented by a vector of length n $(r_i[1], \ldots, r_i[n])$ where $r_i[j]$ is defined as follows. Define $r_i[j] = 0$ if $j \notin [s^i_t, f^i_t]$ for any $t, 1 \leq t \leq k_i$; otherwise $r_i[j] = min\{j - s^i_t + 1, f^i_t - j + 1\}$. We call such a vector, an *interval encoding*. Intuitively, $r_i[j]$ represents how far away in r_i the j^{th} bin is from a white space bin. This gives us a fixed length representation that encodes both the position and the width of the individual block intersections with the i^{th} row. The method defined by computing the Manhattan distance between interval encodings will be called the *interval distance* and will be denoted by d^I.

The effectiveness of interval distance is demonstrated using the examples in Figure 2. Four rows are shown and there are eleven bins per row. The text bins are shown as black and the white space bins are shown as white. For $1 \leq i \leq 4$, the interval encoding $E(row_i)$ of row_i in Figure 2 are given by

$$
\begin{aligned}
E(row_1) &= (1, 2, 3, 4, 3, 2, 1, 0, 1, 2, 1) \\
E(row_2) &= (1, 2, 3, 3, 2, 1, 0, 1, 2, 2, 1) \\
E(row_3) &= (1, 2, 3, 4, 5, 6, 5, 4, 3, 2, 1) \\
E(row_4) &= (1, 0, 1, 2, 3, 4, 5, 4, 3, 2, 1).
\end{aligned}
$$

In the case shown in the figure, row_1 and row_2 are more similar to each other than either is to row_3 since both row_1 and row_2 intersect two blocks at roughly the same horizontal positions whereas row_3 intersects only one very large block. This is reflected by the fact that the edit distance gives $d^E(row_1, row_2) = 2$, much smaller than either $d^E(row_1, row_3) = 7$ or $d^E(row_2, row_3) = 9$. Using the interval distance the results again agrees with perception: $d^I(row_1, row_2) = 6$, while $d^I(row_1, row_3) = 16$ and $d^I(row_2, row_3) = 18$. However, if we simply use d^M, the Manhattan distance between straightforward bitmap representations (i.e., set $r_i[j]$ to 1 if it is a text bin and 0 otherwise), we get $d^M(row_1, row_2) = 2$, larger than $d^M(row_1, row_3) = d^M(row_2, row_3) = 1$. Similarly, row_1 and row_2 are more similar to each other than either is to row_4. Again, the edit distance and interval distance agree with this sense of similarity since $d^E(row_1, row_4) = d^E(row_2, row_4) = 10 > d^E(row_1, row_2) = 2$ and $d^I(row_1, row_4) = 18 > d^I(row_2, row_4) = 17 >$

$d^I(row_1, row_2) = 6$. However, using d^M says that row_1, row_2 and row_3 are each at distance 2 from one another.

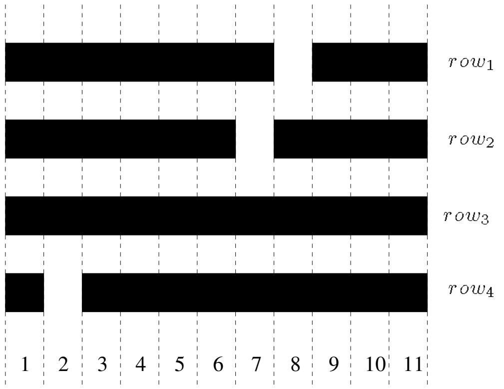

Figure 2. Four examples of rows.

Given a collection $E = \{e_1, \ldots, e_t\}$ of interval encodings, we now discuss how they can be partitioned into clusters whose cluster centers can be easily computed. The interval encodings are vectors of length n. A simple and effective means for computing clusters is the k-means clustering technique [3]. This iterative technique requires computing cluster centers for each candidate cluster and for each e_i computing the nearest cluster center to e_i. We define the Manhattan distance to be the distance measure used for these computations. In this case, a cluster center c is defined to be an n-dimensional vector that minimizes the sum of the Manhattan distances from c to each of the elements of the cluster. It can be shown that the cluster center c of C is an n-dimensional vector whose i^{th} component is the median of the i^{th} component of the elements of C. Note that in general, c will not necessarily be an interval encoding. Thus unlike in the case of the edit distance described above, we can compute a true cluster center since we are not restricted to choosing some element of a cluster to act as the cluster center.

The above discussion shows that given a collection of interval encodings we can use k-means clustering (based on the Manhattan distance measure) to compute clusters and their centers. This leads to a third method for computing the distance between two rows based on their interval encodings. The *cluster distance*, $d^C(r_i, r_j)$, between interval encodings r_i and r_j is defined to be the l_1 distance between c_1 and c_2 where c_1 is the cluster center nearest (in the l_1 sense) to r_i and c_2 is the cluster center nearest to r_j.

3 Experiments

Experiments were conducted using samples of one column letters, two column letters, one column journal pages, two column journal pages and magazine articles, as shown in Figure 1. Examples of two column letters are usually seen in business type letters where one narrow column often consists of lists of items such as names of partners in a law firm or the company's locations, and the other column represents the body of the letter. All the samples were monochrome page images at 300 dpi. The parameters m (number of vertical partitions) and n (number of horizontal partitions) were set empirically to 50 and 30 respectively.

3.1 Layout Comparison

The objective of this experiment was to rank-order a set of documents in terms of their similarity to a test document. We used ten samples from each of the five classes and thus a total of fifty pages. The experiment consisted of choosing a document at random as a test sample and rank-ordering the remaining documents in terms of its similarity with respect to the test document. We tested the three row distance measures described in the previous section: edit distance, interval distance and cluster distance. In order to get a more concrete idea of how well these measures performed, we also compared them to the naive measure obtained by Manhattan distance between straightforward bitmap encodings of rows.

The edit distance is considered to be the most natural measure of distance and thus provides the "ground truth". The interval distance is an approximation of this measure and the cluster distance an even more extreme approximation. Our experiments showed that neither approximation was too far from the ground truth, and both by far outperformed the naive bitmap distance. One example is shown in Figure 3. The test sample is of a two-column journal type of document, and the top three most similar pages using each of the four distance measures are listed.

3.2 Layout Classification

In this experiment thirty samples in each of the five classes were used to build a top-to-bottom hidden Markov model for that class. The interval encodings from all training samples were divided into thirty clusters, and the cluster centers were used as observations. Each model contained a fixed number of twenty states [4]. A set of eighty-nine samples not used in training were then used in the classification experiment. Each test sample was compared with the five models using the Viterbi algorithm to generate likelihood scores [8]. The experiments were run on an SGI O2 workstation (180 Mhz). It took about 2.5 seconds to classify each document (after page segmentation). The classification accuracy for each of the five classes of documents is shown in Table 1. $Accuracy_1$ refers to the percentage of documents whose correct class appeared as the top-choice (top-one accuracy). $Accuracy_2$ refers to the percentage of documents

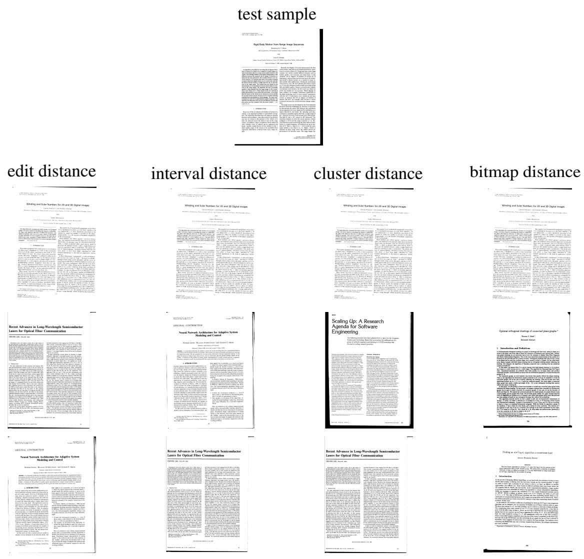

Figure 3. Page layout similarity ranking using different distance measures.

	$Accuracy_1$	$Accuracy_2$
1-col-journal	73	100
2-col-journal	100	100
1-col-letter	70	100
2-col-letter	64	90
Magazine	86	100

Table 1. Percentage accuracy in classification of documents.

whose correct class appeared in either the top or the second choice (top-two accuracy).

As seen from Table 1, the top-two accuracy scores are very high for all but one class, demonstrating the effectiveness of the method as an initial "screening" stage. The accuracies for the two-column letter class are relatively low. One possible reason is that the samples we have collected for that class include many commercial "junk mails" with highly varied layouts and fancy graphic items, causing more errors in segmentation. As a result, the model for that class is relatively weak and thus the two-column letter test samples are more likely to be mis-classified.

4 Conclusions

This paper describes algorithms for visual document similarity comparison based on information related to spatial layout. We introduced a novel feature called interval encoding. This feature was tested using experiments on layout similarity ranking as well as classification. The results in the ranking experiment justify the validity of the features introduced. As seen from the results of the classification experiments, reasonably high accuracy scores are obtained based solely on spatial layout and without OCR. These experiments demonstrate that the features and methods introduced in this paper provide an efficient and effective mechanism for document visual similarity comparison as well as initial classification of documents. Results from this initial analysis can be used by later stages of document processing, including use of class specific geometric and syntactic models incorporating OCR results, to achieve better classification as well as content understanding and extraction.

Acknowledgments

We would like to thank Henry Baird and John Hobby for the numerous helpful discussions.

References

[1] H. S. Baird. Background structure in document images. *Int Journal of Pattern Recognition and Artificial intelligence*, 8(5):1013–1030, 1994.

[2] D. Doermann. The retrieval of document images: a brief survey. In *Proc. ICDAR'97*, pages 945–949, Ulm, Germany, August 1997.

[3] A. Gersho and R. M. Gray. *Vector quantization and signal compression.* Kluwer Academic Publishers, 1992.

[4] J. Hu, R. Kashi, and G. Wilfong. Document classification using layout analysis. In *Proc. First Intl. Workshop on Document Analysis and Understanding for Document Databases*, Florence, Italy, September 1999.

[5] J. J. Hull and J. F. Cullen. Document image similarity and equivalence detection. In *ICDAR'97*, pages 308–312, Ulm, Germany, Aug. 1997.

[6] J. B. Kruskal and D. Sankoff, editors. *Time Warps, String Edits, and Macromolecules: The Theory and Practice of Sequence Comparison*. Addison-Wesley, Reading, MA, 1983.

[7] D. P. Lopresti. Models and algorithms for duplicate document detection. In *Proc. ICDAR'99*, Bangalore, India, September 1999.

[8] L. R. Rabiner and B. H. Juang. *Fundamentals of Speech Recognition*. Prentice Hall, Englewood Cliffs, NJ, 1993.

[9] H. Sakoe and S. Chiba. Dynamic programming algorithm optimization for spoken word recognition. *IEEE Trans. Acoust., Speech, Signal Processing*, ASSP-26:43–49, 1978.

[10] S. L. Taylor, M. Lipshutz, and R. W. Nilson. Classification and functional decomposition of business documents. In *Proc. ICDAR'95*, pages 563–566, Montreal, Canada, August 1995.

[11] W. Zhu and T. Syeda-Mahmood. Image organization and retrieval using a flexible shape model. In *IEEE Int. Workshop on Content Based Access of Image and Video Databases*, pages 31–39, Bombay, India, Jan. 1998.