ZONE CONTENT CLASSIFICATION AND ITS PERFORMANCE EVALUATION

Yalin Wang[†] Robert Haralick[†] Ihsin T. Phillips[‡]

[†] Department of Electrical Engineering
University of Washington Seattle, WA 98195 U.S.A.

[‡] Department of Computer Science, Queens College
CUNY Flushing, NY 11367 U.S.A.

{ylwang@u.washington.edu}

Abstract

This paper presents an improved zone content classification method and its performance evaluation. We added two new features to the feature vector from one previously published method [1]. We assumed different independence relationship in two zone sets. We used an optimized binary decision tree to estimate the maximum zone content class probability in one set while used Viterbi algorithm to find the optimal solution for a zone sequence in the other set. The training, pruning and testing data set for the algorithm include $1,600$ *images drawn from the UWCDROM III document image database. The classifier is able to classify each given scientific and technical document zone into one of the nine classes,* 2 *text classes (of font size* $4-18pt$ *and font size* $19-32\,pt$*), math, table, halftone, map/drawing, ruling, logo, and others. Compared with our previous work [2], it raised the accuracy rate to* 98.52% *from* 97.53% *and reduced the mean false alarm rate to* 0.53% *from* 1.26%.

1 Problem Statement

Let $\mathcal{A}$ be a set of zone entities in a given document page. Let $\mathcal{L}$ be a set of content labels, such as text, table, math, etc. The function $f : \mathcal{A} \to \mathcal{L}$ associates each element of A with a label. The function $V : \mathcal{A} \to \Lambda$ specifies measurements made on each element of $\mathcal{A}$, where Λ is the measurement space.

The zone content classification problem can be formulated as follows: *Given a zone set* $\mathcal{A}$ *and a content label set* $\mathcal{L}$*, find a classification function* $f : \mathcal{A} \to \mathcal{L}$*, that has the maximum probability:*

$$P(f(\mathcal{A})|V(\mathcal{A})) \tag{1}$$

Let $\mathcal{S}_1$ and $\mathcal{S}_2$ be two exclusive and exhaustive subsets of $\mathcal{A}$. We assume the labels of a zone in set $\mathcal{S}_1$ contributes no information relative to the label of another zone in set $\mathcal{S}_2$ and measurements of a zone in set $\mathcal{S}_1$ contribute no information relative to the label of any zone in set $\mathcal{S}_2$. We have

$$\begin{aligned} P(f(\mathcal{A})|V(\mathcal{A})) &= \prod_{i=1,2} P(f(\mathcal{S}_i)|V(\mathcal{A})) \\ &= \prod_{i=1,2} P(f(\mathcal{S}_i)|V(\mathcal{S}_i)) \end{aligned} \tag{2}$$

We have two similar conditions as follows:

1. Conditioned on all the measurements, the label of a zone contributes no information relative to the label of another zone;
2. Measurements of a zone contribute no information relative to the label of another zone.

We assume conditions $1, 2$ are true for set $\mathcal{S}_1$ and only condition 2 holds for set $\mathcal{S}_2$. We also assume the elements of set $\mathcal{S}_2$ are $\{Z_1, Z_2, ..., Z_s\}$, where s is the zone number in $\mathcal{S}_2$. We can have $P(f(\mathcal{S}_1)|V(\mathcal{S}_1)) = \prod_{\alpha\in\mathcal{S}_1} P(f(\alpha)|V(\alpha))$ and

$$\begin{aligned} P(f(\mathcal{S}_2)|V(\mathcal{S}_2)) = {} & P(f(Z_s)|V(Z_s), f(Z_{s-1}), \dots, f(Z_1)) \\ & P(f(Z_{s-1})|V(Z_{s-1}), f(Z_{s-2}), \dots, f(Z_1)) \\ & \dots f((Z_1)|V(Z_1)) \end{aligned}$$

The problem in Equation 2 can be solved by maximizing each individual probability for $\mathcal{S}_1$ and $\mathcal{S}_2$.

In our zone content classification experiment, the elements in set $\mathcal{A}$ are zone groundtruth entities from UWCDROM III document image database [5]. The elements of set $\mathcal{L}$ are text with font size $\leq$ 18pt, text with font size $\geq$ 19pt, math, table, halftone, map/drawing, ruling, logo,

0-7695-1263-1/01/$10.00 © 2001 IEEE

and others. $V(\tau)$ is a feature vector generated for τ, where $\tau \in \mathcal{A}$. Elements of set $\mathcal{S}_1$ are zones in the live-matter part and elements of set $\mathcal{S}_2$ are zones in the header and footer parts in a given page. We used a decision tree classifier to compute each individual probability in set $\mathcal{S}_1$ and use a Hidden Markov Model to model the dependency in set $\mathcal{S}_2$.

2 Related Work and Paper Organization

The zone classification technique plays the key role in the success of a complete document understanding system. Not only is it useful for successive applications such as OCR, table understanding, etc, but it can be used to assist and validate document segmentation.

In the literature, Sivaramakrishnan et. al [1] extracted features for each zone such as run length mean and variance, spatial mean and variance, fraction of the total number of black pixels in the zone, and the zone width ratio for each zone. They used the decision tree classifier to assign a zone class on the basis of its feature vector. They did their experiments on 979 document images from UWCDROM I image database. Liang et. al [3] developed a feature based zone classifier using only the knowledge of the widths and the heights of the connected components within a given zone. Le et. al [4] proposed an automated labeling of zones from scanned images with labels such as titles, authors, affiliations and abstracts. The labeling is based on features calculated from optical character recognition(OCR) output, neural network models, machine learning methods, and a set of rules that is derived from an analysis of the page layout for each journal and from generic typesetting knowledge for English text.

We added two new features: the total area of large horizontal and vertical blank blocks and the number of text glyphs in the given zone, to the original feature vector in a previous paper [1]. We improved the performance of the decision tree by optimizing the trained decision tree. To enrich our model, we incorporated the context constraints to classification for some zones. For some zone set, we modeled their zone class context constraint as a Hidden Markov Model and used Viterbi algorithm [8] to get optimal classification results. We improved the accuracy rate and reduced the false alarm rate for most of the nine classes compared with our previous work [2].

The rest of this paper is divided into 5 sections. Section 3 gives the definitions of two new features. Our improvement of a trained decision tree classifier is given in Section 4. Section 5 describes how we incorporated context constraint to improve classification results using the HMM model. The performance evaluation protocol and experimental results are reported in Section 6. Our conclusion and statement of future work are discussed in Section 7.

3 Two New Features

We use white-pixels to represent background pixels and black-pixels for foreground pixels.

Definition 1: Let $\mathcal{Z}$ represent a *zone* with R rows and C columns. Let (x_1, y_1) be the coordinate of its lefttop vertex, $\mathcal{Z} = \{(r,c) \in Z \times Z | x_1 \leq r < x_1 + R, y_1 \leq c < y_1 + C\}$.

Definition 2: Let p be a *horizontal white run p*, $p = ((r_1, c_1), ..., (r_n, c_n))$, where $(r_i, c_i) \in \mathcal{Z}$, $r_i = r_{i-1}$, $c_i = c_{i-1} + 1$, for $i = 2, ..., n$, and pixel (r_1, c_1), (r_n, c_n) must have a black pixel or the zone border on its left and right side, respectively. For each run, we call the location of the starting pixel of the run and its horizontal length as Row(p), Column(p), and Length(p), respectively.

Definition 3: Let $\mathcal{HR}$ be a *horizontal blank block*, $\mathcal{HR} = (p_1, ..., p_n)$, where Row($p_i$) = Row($p_{i-1}$) + 1, Column($p_i$) = Column($p_{i-1}$), Length($p_i$) = Length($p_{i-1}$), for $i = 2, ..., n$. Clearly, the same idea can be applied to define *vertical white run* and *vertical white blank block*, $\mathcal{VR}$.

Definition 4: A horizontal blank block $\mathcal{HR} = b_r \times b_c$, with lefttop vertex coordinate (x_{b1}, y_{b1}), is a *large horizontal blank block* if and only if it satisfies the following conditions:

1. $\frac{b_c}{C} > \theta_1$, where θ_1 is 0.1;
2. $x_{b1} \neq x_1$ and $x_{b1} + b_c \neq x_1 + C$, where C is the column number in the zone.

Definition 5: A vertical blank block $\mathcal{VR} = b_r \times b_c$, with lefttop vertex coordinate (x_{b1}, y_{b1}), is a *large vertical blank block* if and only if it satisfies the following conditions:

1. $b_r \gg mh$ and $\frac{b_c}{mw} > \theta_2$, where mh and mw are the median height and median width of text glyphs in the zone. θ_2 is empirically determined as 1.4;
2. $x_{b1} \neq x_1$ and $x_{b1} + b_c \neq x_1 + C$, where C is the column number in the zone.

The newly added features are:

1. The total area of large horizontal and large vertical blank blocks, A, $A = \sum_{R \in \mathcal{B}} Area(R)$, where $\mathcal{B} = \{R | R = \mathcal{HR} \text{ or } \mathcal{VR}, \mathcal{HR} \subset \mathcal{Z} \text{ and } \mathcal{VR} \subset \mathcal{Z}\}$;
2. The number of text glyphs in this zone, N, normalized by the zone area, $\frac{N}{Area(\mathcal{Z})}$.

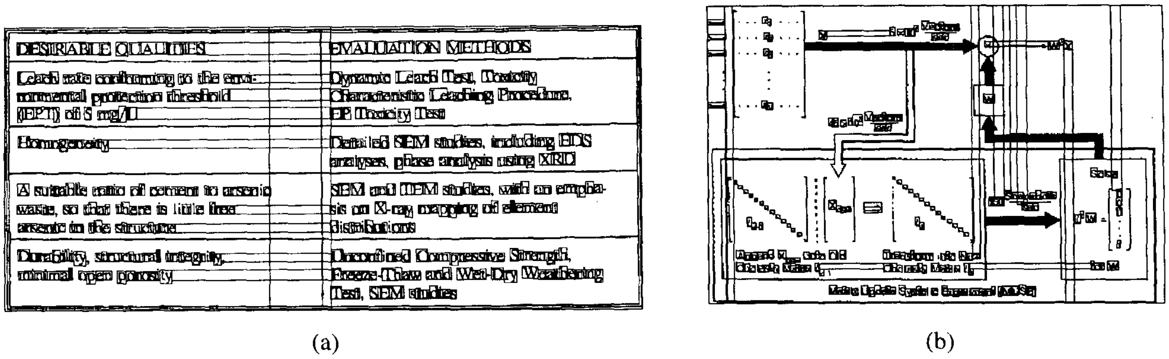

Figure 1. Illustrates the example bounding boxes of large horizontal blank blocks, large vertical blank blocks, and text glyphs. The so-called text glyphs are labeled by a statistical glyph filter. (a) a table zone example; (b) a map/drawing zone example.

The so-called text glyphs are not from any OCR output. They are outputs of a statistical glyph filter. The statistical glyph filter classifies each connected component into one of two classes: text glyph and non-text glyph. Figure 1 illustrates two zone examples showing the overlayed bounding boxes of large horizontal blank blocks, large vertical blank blocks and text glyphs. Together with the 67 features used in Sivaramakrishnan et. al's work [1], our feature vectors have a total of 69 features.

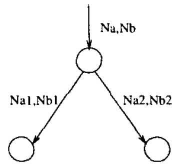

Figure 2. Illustrates a decision tree node with its child nodes

4 Eliminating Data Over-fitting in Decision Tree Classifier

A decision tree classifier makes the assignment through a hierarchical, tree-like decision procedure. For the construction of a decision tree [6], we need a training set of feature vectors with true class labels. At each node, a discriminant threshold is chosen such that it minimizes an impurity value. At each node, the discriminant function splits the training subset into two subsets and generates two child nodes. The process is repeated at each newly generated child node until a stopping condition is satisfied and the node is declared as a leaf node based on a majority vote.

In building a decision tree classifier, there is a risk of memorizing the training data, in the sense that nodes near the bottom of the tree represent the noise in the sample. As mentioned in [7], some methods were employed to make better class probability, such as building multiple trees and use the benefits of averaging, approximate significance tests, etc. We used two simple methods to reduce the data over-fitting in the trained decision tree.

In Figure 2, there is a node with its two child nodes. N_a, N_b are the number of class A vectors and class B vectors which arrive at this node. Similarly, N_{a1} and N_{b1}, N_{a2} and N_{b2} are the number of class A vectors and class B vectors which arrive at its left child node and its right node, respectively. We can compute two different error probabilities associated with this node.

- *Inherent Error Rate.* It is the error probability of setting this node as a leaf node,
$$\frac{min(N_a, N_b)}{N_a + N_b};$$
- *Effective error rate.* It is the error probability of expanding this node with two children nodes,
$$\frac{min(N_{a1}, N_{b1}) + min(N_{a2}, N_{b2})}{N_a + N_b}.$$

The probability ratio is

$$e_I = \frac{min(N_{a1}, N_{b1}) + min(N_{a2}, N_{b2})}{min(N_a, N_b)}$$

The condition is that if $e_I \geq \theta$, where θ is a given threshold, we will make this node as a leaf node, otherwise, we keep its two child nodes.

Another constraint condition is that we will stop expanding this node if the probability of an arbitrary separation of vector numbers is less than a threshold. The probability can be computed as

$$e_2 = \frac{C_{N_a}^{N_{a1}} C_{N_b}^{N_{b1}}}{C_{N_a+N_b}^{N_{a1}+N_{b1}}} = \frac{C_{N_a}^{N_{a2}} C_{N_b}^{N_{b2}}}{C_{N_a+N_b}^{N_{a2}+N_{b2}}}$$

where $C_{N_a}^{N_{a1}}$ is the combination number of the elements of N_a taken N_{a1} at a time. If $e_2 < \delta$, where δ is a given threshold, we will stop expanding this node and make it as a leaf node.

5 Finding the Optimal Sequence in $\mathcal{S}_2$ by HMM

To further improve the zone classification result, we want to make use of context constraint in some zone set, $\mathcal{S}_2$. We model context constraint as a Markov Chain. Let s be element number in $\mathcal{S}_2$. Let $Z = (Z_1, Z_2, ..., Z_s)$, where $Z_t \in \mathcal{S}_2, t = 1, ..., s$, be a zone sequence. We have

$$\begin{aligned} &P(f(Z_t)|V(Z_t), f(Z_{t-1}), ..., f(Z_1)) \\ &\quad = P(f(Z_t)|V(Z_t), f(Z_{t-1})) \end{aligned} \tag{3}$$

We can use Viterbi algorithm([8]) to find the most likely state sequence Z^*, which can be used as the optimal solution to Equation 3. To apply Viterbi algorithm([8]), we have to know the probability that each zone belongs to each class. This probability is readily estimated from the training data set by decision tree structure. Suppose example x falls to leaf l in the tree structure T. We have C mutually exclusive and exhaustive classes, $d_1, ..., d_C$. A vector ϕ_l is associated with the leaf node l. Its elements are the proportion of the number of class d_i training samples over the number of total training samples falling to leaf l. We can compute probability that x belongs to each class by

$$P(c = d_j|x, T) = \phi_{l,j}, j = 1, .., C.$$

Because of the biased data, we applied HMM on header and footer regions instead of the whole page. Of a total of $24,177$ zones, there are $21,512$ text 1 class zones. The zone numbers of table class, halftone class and map/drawing class are 215, 388 and 710, respectively. Most table, halftone and map/drawing zones are followed by a text zones. When we apply HMM on the whole page, it tends to recognize table zones as map/drawing zones since in the HMM training data set the number of map/drawing zones is far larger than that of table zones. In the header/footer regions, there are only text 1 class zones, text 2 class zones, rule class zones and others class zones. HMM solution gives us about 0.48% improvement in $2,274$ zones of header and footer regions. Although the improvement is very limited, we claim HMM would give us more improvement if the data are less biased.

6 Experiments and Results

A hold-out method is used for the error estimation in our experiment. We divided the data set into 9 parts. We trained the decision tree on the first 4 parts, pruned the tree using another 4 parts. and then tested on the last 1 part. To train the Markov model, we trained on the first 8 parts and tested it on the last 1 part. Continue this procedure, each time omitting one part from the training data and then testing on the omitted part. Then the combined 9 part results are put together to estimate the total error rate [6].

The output of the decision tree is compared with the zone labels from the ground truth in order to evaluate the performance of the algorithm. A contingency table is computed to indicate the number of zones of a particular class label that are identified as members of one of nine classes. The rows of the contingency table represent the true classes and the columns represent the assigned classes. The cell at row *r* and column *c* is the number of zones whose true class is *r* while its assigned class is *c*. We compute four rates here: *Correct Recognition Rate(CR), Mis-recognition Rate(MR), False Alarm Rate(FR), Accuracy Rate(AR).* Suppose we only have two classes: a and b. The possible true- and detected-state combination is shown in Table 1. We compute four rates for class a as follows:

True	Assigned Class	
Class	a	b
a	P_{aa}	P_{ab}
b	P_{ba}	P_{bb}

Table 1. Possible true- and detected-state combination for two classes

$$\begin{aligned} CR &= \frac{P_{aa}}{P_{aa} + P_{ab}} \\ MR &= \frac{P_{ab}}{P_{aa} + P_{ab}} \\ FR &= \frac{P_{ba}}{P_{ba} + P_{bb}} \\ AR &= \frac{P_{aa} + P_{bb}}{P_{aa} + P_{ab} + P_{bb} + P_{ba}} \end{aligned} \tag{4}$$

In our experiment, the training and testing data set was drawn from the scientific document pages in the University of Washington document image database III [5]. It has $1,600$ scientific and technical document pages with a total of $24,177$ zones. The class labels for each of the zones are obtained from the database. These zones belonged to nine different classes. For a total of $24,177$ zones, the accuracy rate was 98.52% and mean false alarm rate was 0.53%. The

	T1	T2	M	T	H	M/D	R	L	O	CR	MR
T1	21446	13	34	10	2	3	2	1	1	99.69%	0.31%
T2	16	106	1	0	2	1	1	0	1	82.81%	17.19%
M	49	2	683	3	0	17	1	2	1	90.11%	9.89%
T	9	0	4	159	1	38	1	1	2	73.95%	26.05%
H	1	2	0	1	369	12	0	1	2	95.10%	4.90%
M/D	8	0	20	30	18	630	1	1	2	88.73%	11.27%
R	6	0	2	0	1	3	419	0	0	97.22%	2.78%
L	5	5	0	1	0	1	1	0	0	0.00%	100.00%
O	3	2	1	2	4	3	0	0	7	31.82%	68.18%
FR	3.64%	0.10%	0.26%	0.20%	0.12%	0.33%	0.03%	0.02%	0.04%		

Table 2. Contingency table showing the number of zones of a particular class that are assigned as members of each possible zone class in UWCDROM III. In the table, T_1, T_2, M, T, H, MD, R, L, O represent text with font size $\leq$ 18pt., text with font size $\geq$ 19pt., math, table, halftone, map/drawing zone, ruling, logo, others, respectively.

results are better than our previous result [2], which were 97.53% and 1.26%.

7 Conclusion and Future Work

Given the segmented document zones, correctly determining the zone content class is very important for the further processes. We improved a zone classification method by adding two new features, pruning the decision tree and modeling context constraints as HMM for some zone set. In 1,600 UWCDROM III images, our zone classification method can classify each given zone into one of the nine classes. Compared with our previous work, we raised the accuracy rate to 98.52% from 97.53% and reduced the mean false alarm rate to 0.53% from 1.26%. Our future work will include incorporating context constraint in set $\mathcal{A}$, studying our zone classifier performance on a even larger image data set.

References

[1] R. Sivaramakrishnan, I. Phillips, J. Ha, S. Subramanium, and R. Haralick: Zone classification in a document using the method of feature vector generation. Proceedings of the 3rd ICDAR, pages 541–544, Aug. 1995.

[2] Y. Wang, R. Haralick, and I. T. Phillips: Improvement of zone content classification by using background analysis. DAS2000, Rio de Janeiro, Brazil, 10-13 December, 2000.

[3] J. Liang, R. Haralick, and I. T. Phillips: Document zone classification using sizes of connected components. Document Recognition III, SPIE'96, pages 150–157, 1996.

[4] D. X. Le, J. Kim, G. Pearson, and G. R. Thom: Automated labeling of zones from scanned documents. Proceedings SDIUT99, pages 219–226, 1999.

[5] I. Phillips: Users' reference manual. CD-ROM, UW-III Document Image Database-III, 1995.

[6] R. Haralick and L. Shapiro: Computer and Robot Vision. Addison Wesley, Vol 1, 1992.

[7] W. Buntine: Learning Classification Trees. Statistics and Computing journal, Vol. 2, pages 63-73, 1992

[8] L.R. Rabiner: A tutorial on hidden Markov models and selected applications in speech recognition. Proceedings of the IEEE pp 257–285, Vol. 77, No. 2, February, 1989.