A Hypothesis Testing Approach to Word Recognition Using Dynamic Feature Selection

Liang Li, Tin Kam Ho,
Jonathan J. Hull, Sargur N. Srihari
Center for Document Analysis and Recognition
State University of New York at Buffalo
Buffalo, NY 14260, USA

Abstract

A top-down approach to word recognition is proposed. Discussions are presented on dynamically selecting the most effective feature combinations, which are applied to discriminate between a limited set of word hypotheses.

1 Introduction

Word recognition is traditionally accomplished with a bottom-up strategy. A set of features and their detection operators are first designed. All the operators are applied to an input image to detect the desired features. The feature descriptors are then matched to those derived from the class prototypes to compute a similarity score. The class identity is then decided based on the similarity scores. Essentially, all the classes are considered simultaneously in a one-pass discrimination process.

An alternative approach is to focus the discrimination on a subset of classes at a time. Recognition is achieved in a multiple of stages. In each stage, a set of class hypotheses is proposed and tested. Final recognition is achieved by combining the results of the individual tests [1]. The advantage of this method is that it allows a dynamic selection of the most effective features for discrimination between a particular set of hypotheses. This is a top-down strategy, where feature detection and matching are dynamically guided by the hypotheses.

This method is different from a decision-tree approach [2] in that the recognition procedure is not guided by the results of previous comparisons. Each individual test is independently conducted, and the features used in each test are completely determined by the particular hypotheses and related training data.

The difficulties in this approach include how to generate the hypotheses, how to determine a set of features that are most effective for discrimination between a particular set of hypotheses, and how to combine the results of individual tests on different hypotheses. In this paper, we discuss our solutions to these problems, in the context of an application to the recognition of machine-printed words.

2 Hypothesis Testing in Word Recognition

Word recognition is concerned with identifying a word image as one of a set of words contained in a given lexicon. Many previous approaches have been proposed, including character recognition based methods and wholistic approaches which treat a word as a single symbol [6]. However, it is difficult to simultaneously discriminate between all the words in a large lexicon, such as one containing 100,000 words, especially when the input image is noisy. Typically, a technique is able to include the true word in a set of guesses (referred to as a *neighborhood* of the input word) with a high probability, but not uniquely identify it. A method is needed that will further discriminate between the words in the neighborhood [7].

In this context, the *hypotheses* or *hypothesis set* are the words in a neighborhood. Since the neighborhoods are computed by some recognition techniques, they usually contain words that are similar in shape, especially if a very large lexicon was used originally. An example of such a neighborhood is {CANFA, CANTY, DANTE, SANTA, SANTO} for a word 'SANTA'.

Based on these observations, we assume that the set of hypotheses is small, and the hypothesized words may share some similarities in shape. Since words with very dissimilar shapes are easy to discriminate by a traditional method, they do not need to be discriminated by hypothesis testing. Nevertheless, the proposed method is applicable even if the words are not similar.

Suppose that a small lexicon is given as hypotheses for an input word image. To identify the true word, discrimination should be focused on the differences between the hypothesized words. In order to concentrate on a minimum number of differences, two word hypotheses are discriminated at a time. A lexicon of m words therefore needs $m(m-1)/2$ tests. This implies an $O(m^2)$ algorithm, yet it is feasible as long as the hypothesis set is small.

To discriminate between two words, a feature comparison is needed only for the distinct characters in the two words. For example, to distinguish between CANFA and SANTA, it is needed to determine whether the first character in the input word is C or

S, and the fourth character is F or T. In the next section, we discuss how to select a set of features for discrimination between two characters.

3 Selecting Features for Discrimination Between Two Characters

Assume a set $S_N = \{x_1, ..., x_N\}$ of well defined features are available. The feature selection problem can be stated as follows: For a given discrimination problem, compute $S_n = \{x'_1, ..., x'_n\}$ ($S_n \subseteq S_N$) which is the most effective in solving the problem in accordance with a given feature evaluation criterion.

The problem is nontrivial since ordering single features by their individual effectiveness is insufficient [4] [8]. Clearly, for a given problem, the optimal S_n can be obtained by an exhaustive search on all subsets of S_N. It is stated in [4] that this is the only way to solve the problem. However, this is infeasible, if not impossible, even for moderate N, since there are 2^N possible S_n's. An alternative is to achieve sub-optimality using a feasible selection procedure.

The linear search approach is a simple solution. The intuitive idea is that the best set of n features probably includes the best set of n-1 features (though it is known that the claim is not always true [5]). It works as follows.

Let $S_{t_n} = \{x^n_{t_1}, ..., x^n_{t_n}\}$ be the best set of n features obtained at the n^{th} stage ($S_{t_n} \subseteq S_N$). The linear search approach first selects the best single feature from S_N according to certain criterion. Then it selects the best single feature from the remaining set, appends it to the previous one, and evaluates the effectiveness of the combination on some training data. This selection, appending, and evaluation procedure continues until some predefined conditions are satisfied, such as no more improvement in the effectiveness of S_{t_n} are made by adding more features, or a predetermined subset size has been reached, or the given discrimination problem is perfectly solved using S_{t_n}.

Since this is an $O(n)$ algorithm, this approach has the advantage that very few sets of features need to be evaluated. The disadvantage is that the feature set thus obtained contains all the previously selected best single features. As mentioned earlier, the collection of the best single features is not necessarily the most effective for a given problem.

Chang [3] proposes a dynamic programming approach for searching for a sub-optimal feature combination. It, on one hand, significantly reduces the number of feature subsets to be evaluated to compute the best feature combination. On the other hand, it also considers combinations which may not include all the best single features selected in the previous stages. A generalized dynamic programming approach for this problem is described as follows.

Let $E_n(x_1, ..., x_n)$ be an evaluation function on the effectiveness of $\{x_1, ..., x_n\}$ according to some feature selection criterion. $O_n(x^n_{t_1}, ..., x^n_{t_n})$ is the maximized $E_n(.)$ at the n^{th} stage. $T_n(.)$ is a transformation function which yields a set of n features according to its arguments. The recursive formula for computing the best feature subset at the n^{th} stage with a given k (≥ 1) is

$$O_n(S_{t_n}) = \max_{x_i \in S_N} E_n(T_n(x_i, S_N, S_{t_{n-1}}, k))$$

where (let $\overline{S_{t_{n-1}}} = S_N - S_{t_{n-1}}$)
$T_n(x_i, S_N, S_{t_{n-1}}, k)$

$$= \begin{cases} S_{t_{n-1}} \bigcup \{x_i\}, & if\ x_i \in \overline{S_{t_{n-1}}}; \\ \{x_{t_1}, ..., x_{t_k}, x_{t_{k+1}}, ..., x_{t_n}\}\ such\ that \\ O_{n-k}(x_{t_{k+1}}, ..., x_{t_n}) = \\ \max_{x_j \in \overline{S_{t_{n-1}}}} E_{n-k}(T_{n-k}(x_j, \overline{S_{t_{n-1}}}, S_{t_{n-k-1}}, k)), \\ \qquad if\ \{x_{t_1}, ..., x_{t_k}\} \subset S_{t_{n-1}}, \\ \qquad and\ (k+1) < n \leq \frac{(N+k+1)}{2}; \\ \emptyset, \qquad otherwise. \end{cases}$$

In order to construct S_{t_n}, the algorithm first constructs a candidate set of n features by selecting each single feature from the complement of $S_{t_{n-1}}$ and appending it to $S_{t_{n-1}}$. Then the algorithm keeps any combinations of k features from $S_{t_{n-1}}$, selects the best set of $n-k$ features from the complement of $S_{t_{n-1}}$ by recursively using the same approach, and constructs a candidate set of n features. This step demonstrates that the best set of n features selected may not include all the best single features selected in the previous stages. Choosing different values of k gives different variants of this approach. $k = n-2$ is a special case described in [3]. If the upper bound of the set size is N, then at most

$$\sum_{i=1}^{N} \begin{pmatrix} N-i+1 \\ 1 \end{pmatrix} + \sum_{i=2}^{N-1} \begin{pmatrix} (N-i+1)-1 \\ 1 \end{pmatrix} + \sum_{i=3}^{N-1} \begin{pmatrix} i-1 \\ i-2 \end{pmatrix} = \frac{(3N-2)(N-1)}{2}$$

feature subsets need to be evaluated to construct the optimal subset. Hence, this is an $O(N^2)$ algorithm in this case.

Compared with the linear search approach, the dynamic programming approach considers more general cases, yet it is still much less expensive than an exhaustive search.

In an application to word hypothesis testing, the objective is to obtain the most effective feature combinations, when given S_N, in discriminating between each possible pair of characters (C_1, C_2). 62 character classes are considered, including 26 letters in upper and lower case, and 10 numerals. Hence there are totally $(62 \times 61)/2 = 1,891$ possible pairs of character classes.

The effectiveness of a particular S_{t_n} is measured with a training set, which contains samples of C_1 and C_2, and a set of prototypes of each class. The effectiveness of S_{t_n} on discriminating between C_1 and C_2

can be evaluated by either maximizing the number of correct classifications, or maximizing a distance measure, on some training data.

This algorithm is applied to each of the 1,891 pairs of character classes. This results in 1,891 different feature subsets. Each of these feature subsets is used whenever discrimination is needed between the corresponding character pair.

4 Discriminating Between Two Word Hypotheses

We assume that the lengths of the two hypothesized words are the same or differ by at most one character. Since words of very different lengths, such as 'UNIVERSITY' and 'HILL', can be discriminated using an estimate of the word length, this assumption is not a major limitation.

To discriminate between two word hypotheses of the same length, the distinct characters in the two words are first located. The feature subsets corresponding to these distinct characters are selected. Hence the feature sets used for discrimination are dynamically adapted to different hypotheses. The selected features are extracted from the prototype images of the two words. A descriptor is constructed for each hypothesized word using these features.

An input word image is first segmented into individual characters. Only the character images corresponding to the positions of the distinct characters in the two hypothesized words need to be processed. For example, to determine whether an input image is 'CANFA' or 'SANTA', only the first and the fourth characters need to be processed. Features contained in the previously selected subsets are extracted from these character images. A descriptor of the image is then constructed using the extracted features. It is matched to the descriptors of the hypothesized words. The image is then determined to be closer to either of the two words by a distance measure defined on the selected features.

This procedure is applied more than once if the words are of different lengths. The lengths of the words are adjusted to be the same by removing one character before each application. The distance from an image to a hypothesized word is taken as the minimum of the distances computed between all adjusted versions of the image and the word. The image's identity is determined to be in favor of either of the two hypotheses by the distances.

5 Combining Results of Individual Tests on a Lexicon

A lexicon of m words needs $m(m-1)/2$ individual tests. Each word in the lexicon is tested against $m-1$ other words. The number of times it is favored in these tests is counted. All the words are then ranked according to these counts. This ranking will be used later to determine the identity of the image with respect to the lexicon.

If all the counts are distinct, then the ranking has no ambiguity. Otherwise, there are ties that need to be resolved. A method for resolving the ties is given as follows.

Let $G = (V, E)$ be a complete digraph. Each vertex $v \in V$ corresponds to a word in the lexicon. There is a directed edge $e \in E$ between every two vertices. The direction of an edge represents the result of a pairwise test. According to the result of the test, the edge is directed from the vertex corresponding to the favored word to the vertex corresponding to the unfavored word. Therefore, the number of times that a word is favored equals to the out-degree of the corresponding vertex in G.

There are no ties in the out-degrees of vertices if G is an acyclic, complete digraph. Otherwise, there exists $V_1 \subseteq V$ such that all v in V_1 have the same out-degree. This ambiguity can partly be resolved by restricting the counts of the out-degrees to a subgraph on V_1, as described in the followings.

Construct a subgraph $G_1 = (V_1, E_1)$, where V_1 contains a set of vertices that have the same out-degree in G, and E_1 contains those edges in G that connect the vertices in V_1. Recompute the out-degrees in G_1. If all the recomputed out-degrees are distinct, then order the vertices in V_1 by these out-degrees. Otherwise, recursively apply the same procedure to the vertices with the same out-degree in G_1. The ranking within G_1 is then embedded to G_1's original position in G. An example is given in Figure 1.

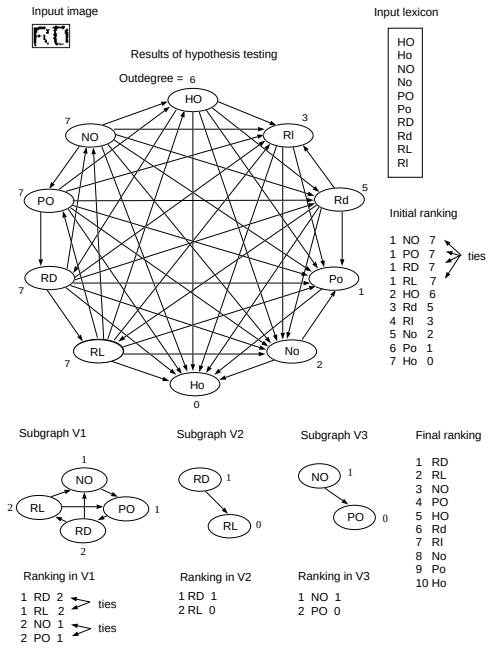


Figure 1: Resolving rank ambiguities by the subgraph method.

This procedure does not guarantee that all the ties in the out-degrees will be resolved. This is shown in

the following. Suppose $|V_1| = n$. Since $G_1 = (V_1, E_1)$ is a complete digraph, therefore, $|E_1| = n(n-1)/2$. Assume that $O(v)$ is the out-degree of v, $v \in V_1$. We have

$$\sum_{v \in V_1} O(v) \ = \ |E_1| = \frac{n(n-1)}{2} \tag{1}$$

Suppose $O(v_1) = O(v_2)$, $\forall v_1, v_2 \in V_1$. From (1), we know that

$$O(v_1) = O(v_2) = \frac{\sum_{v \in V_1} O(v)}{n} = \frac{n-1}{2} \tag{2}$$

Case 1: n is even. Since n is even, then $O(v) = (n-1)/2$ cannot be an integer for $v \in V_1$. This contradicts the definition of an out-degree. In other words, if the number of vertices with the same out-degree in the original graph is even, then the out-degrees of these vertices in the subgraph will not be all the same, and hence the problem is reducible in this case.

Case 2: n is odd. Therefore $n-1$ is even, and $O(v) = (n-1)/2$ is an integer for $v \in V_1$. That means, if there is an odd number of vertices sharing the same out-degree in the original graph, their out-degrees in the subgraph may also be the same. Hence the problem is not necessarily reducible in this case.

The out-degree for each vertex does not contain the information about how strongly the corresponding hypothesis is favored. Therefore, in addition to counting the number of times each word is favored among all pairwise tests, a distance measure is computed at the same time for each word as the sum of the distances of that word to the image in all pairwise tests. This distance measure is normalized with respect to the number of features used. The final ranking among the words in the lexicon can then be determined by combining the out-degree based ranks and the distance based ranks.

6 Experimental Results

An experiment was performed using pixel values as features for discriminating between degraded machine-printed characters. Each character was normalized to 24×24 pixels, and then divided into 4×4 regions, each of which was taken as a feature. The dynamic programming approach given $k = n - 2$ was applied in constructing $1,891$ feature subsets for all possible character pairs with a training set containing $22,265$ character images. The constructed subsets were then tested on a separate test set which contains $5,449$ character images. 751 out of all the $1,891$ pairs were better discriminated using the obtained subsets, as compared to those when using the original set of 16 features. 426 pairs got worse results. The average size of the subsets was reduced from 16 to 6.82.

The selected features were then used to recognize a set of 923 word images extracted from envelope images collected from live mail. The words are machine-printed in multiple font types, and are of variable quality. For each image, a neighborhood of up to 10 words was given as hypotheses. The neighborhood was computed using a bottom-up recognition technique [6]. The average size of the neighborhoods is 8.98 words. Table 1 shows the results of using feature subsets and a whole set for recognizing those 923 word images.

Table 1: Applying hypothesis testing on recognizing 923 machine-printed word images

Correct in top N choices, N=	1	2	5
using whole feature set (16 features) (%)	91.0	96.4	98.9
using subsets constructed by dynamic programming approach, when k=n-2 (%)	93.9	96.9	99.3

7 Conclusion

A hypothesis testing approach is proposed for word recognition. It allows a dynamic selection of the most effective and minimum size feature set for discrimination between a set of hypothesized words. Selecting a set of features according to their effectivenesses is discussed. The success of the approach was demonstrated in an application to the recognition of degraded machine-printed images.

References

[1] C.A. Becker, Semantic Context and Word Frequency Effects in Visual Word Recognition, *Journal of Experimental Psychology: Human Perception and Performance 9*, 5, 1979, 726-738.

[2] R.G. Casey, G. Nagy, Decision Tree Design Using a Probablistic Model, *IEEE Transactions on Information Theory*, Vol. IT-30, No. 1, January 1984, 93-99.

[3] C.Y. Chang, Dynamic Programming as Applied to Feature Subset Selection in a Pattern Recognition System, *IEEE Transactions on Systems, Man, and Cybernetics*, Vol. SMC-3, No. 2, March 1973, 166-171.

[4] T.M. Cover, Computational Complexity Aspects of Dimensionality Reduction and Classification, *IEEE Transactions on Systems, Man, and Cybernetics*, Vol. SMC-1, October 1971, 402.

[5] T.M. Cover, The Best Two Independent Measurements Are Not the Two Best, *IEEE Transactions on Systems, Man, and Cybernetics*, January 1974, 116-117.

[6] T.K. Ho, J.J. Hull, S.N. Srihari, Word Recognition with Multi-Level Contextual Knowledge, *Proceedings of the First International Conference on Document Analysis and Recognition*, Saint-Malo, France, 1991, 905-915.

[7] J.J. Hull, Hypothesis Testing in a Computational Theory of Visual Word Recognition, *Proceedings of the Sixth National Conference on Artificial Intelligence (AAAI)*, Seattle, Washington, July 13-17, 1987, 718-722.

[8] S.N. Srihari, On Choosing Measurements for Invariant Pattern Recognition, *Information Sciences*, 21(1), 1980, 1-11.