A Hidden Markov Model for Language Syntax in Text Recognition

Jonathan J. Hull
Center of Excellence for Document Analysis and Recognition
Department of Computer Science
State University of New York at Buffalo
Buffalo, New York 14260 USA
hull@cs.buffalo.edu

Abstract

The use of a hidden Markov model (HMM) for language syntax to improve the performance of a text recognition algorithm is proposed. Syntactic constraints are described by the transition probabilities between word classes. The confusion between the feature string for a word and the various syntactic classes is also described probabilistically. A modification of the Viterbi algorithm is also proposed that finds a fixed number of sequences of syntactic classes for a given sentence that have the highest probabilities of occurrence, given the feature strings for the words. An experimental application of this approach is demonstrated with a word hypothesization algorithm that produces a number of guesses about the identity of each word in a running text. The use of first and second order transition probabilities is explored. Overall performance of between 65 and 80 percent reduction in the average number of words that can match a given image is achieved.

1. Introduction

Text recognition algorithms often process only images of isolated characters. This is sometimes followed by a post-processing step that uses information from a dictionary of allowable words to correct recognition errors. This approach can provide high performance for good quality images. For example, a 99.5 percent correct character recognition rate has been reported for commercial character recognition devices on clean images. However, even this level of performance still implies that a typical page of text containing about 4000 characters or approximately 800 words would still contain about 20 errors. This performance would be much worse if degraded images were input such as facsimile documents or multiple generation photocopies.

A computational model for word recognition has been proposed that overcomes some of the constraints of other methodologies [3]. The design of this technique is based on human performance in reading which suggests that the feature analysis of word images includes a wholistic analysis of word shape. Also, feature extraction from a word image is only one part of a complex process of understanding a text.

One part of the understanding process that underlies word recognition is an analysis of the syntax of the text. Language syntax has been modeled by the binary constraints between a group of words with the same shape and the syntactic classes that could follow them [4]. The constraints were compiled from a training text and applied to restrict the decisions for the syntactic class of a word to be consistent with the shape of the previous word. Even this limited information was shown to be effective at reducing the average number of words that could match any image by about 16 percent on average with an error rate of about one percent. An error occurred when a word was erroneously removed from consideration.

This paper proposes to model English syntax as a Markov process where the probability of observing any syntactic class tag is dependent on the class of the previous word. This model is applied to text recognition by first determining a number of alternatives for the identity of each word. The syntactic tags of the alternatives for the words in a sentence are then input to a modified Viterbi algorithm that determines a fixed number of sequences of syntactic classes that include each word. An alternative for a word decision is output only if its syntactic class is included in at least one of these sequences. Word recognition performance is improved if the number of alternatives is reduced and the correct choice is not removed.

0-8186-2915-0/92 $3.00 © 1992 IEEE

2. Text Recognition Algorithm

The recognition algorithm that incorporates the Markov model of syntax contains three steps [3]. Word images are input sequentially. First, a group of words in a dictionary that are visually similar to an input image are calculated by a *hypothesis generation* algorithm. This group of dictionary words is referred to as the *neighborhood* of the input word. Next, in a step called *global contextual analysis*, language syntax is used to refine the contents of the neighborhood. Words are removed from neighborhoods if their syntactic classes are inconsistent with surrounding words in the text. In a final step called *hypothesis testing*, the contents of the reduced neighborhood are used to focus specialized feature testing routines that identify the word in the image.

3. Syntax Model

The syntax of a sentence is summarized as the sequence of syntactic classifications for its words. A specific sequence of syntactic classes is referred to as a "parse" for the sentence. For example, in the sentence "He was at work.", *He* is a pronoun, *was* is a verb, *at* is a preposition, and *work* is a noun. Since it is known that the appearance of any syntactic class probabilistically constrains the classes that can follow it, a Markov model is a natural representation for syntax [6].

A hidden Markov model (HMM) can be specified that links the recognition process described earlier and a Markov model for language syntax [7]. The syntactic classes in the English language are assumed to be the N *states* of a discrete n^{th} order Markov process. In the word recognition algorithm, the states are "hidden" because they are not observable at run-time. Rather, the feature vector that describes a word is the observable event. The number of such events is finite and provides a fixed number M of *observation symbols*.

The transition from one state to another is described by a *state transition probability distribution*. There is also a probabilistic constraint on the appearance of an observation or feature vector given that the model is in a specific state. This constraint is sometimes referred to as the *confusion probability*. There is also an *initial state distribution* that specifies the state or syntactic class that the model is in for the first word in a sentence.

The HMM is completely specified by the five elements just described (states, observation symbols, state transition probabilities, observation symbol probabilities, and initial probabilities). The HMM is applied to word recognition by estimating the sequence of states (syntactic classes) with the maximum a-posteriori probability of occurrence for a given sequence of observations (feature vectors). Each observation can correspond to a number of different words with a similar feature vector (i.e., the neighborhood in the word recognition algorithm). The performance of word recognition is improved by removing words from neighborhoods that have syntactic classes which do not occur on the estimated state sequence.

The estimation of the sequence of states with the maximum a-posteriori probability of occurrence is efficiently performed by the Viterbi algorithm [1]. The adaptation of the Viterbi algorithm to this problem is very similar to its use in postprocessing character decisions [2].

A useful modification of the Viterbi algorithm is to allow it to find a fixed number of syntactic class sequences (parses) for a sentence, each of which have the next best cost among all possible alternatives. This modification is simply implemented by maintaining the desired number of alternative sequences and their costs at each point in the evaluation. The final result includes the parses and their costs.

4. Experimental Investigation

Experimental tests were conducted to determine the ability of the HMM to reduce the number of word candidates that match any image. Given sentences from test samples of running text, a set of candidates for each word were produced by a model for the hypothesis generation portion of the word recognition algorithm. These candidates were looked up in a dictionary to retrieve their syntactic classes as well as their confusion probabilities. The Viterbi algorithm was then run on these data, using transition probabilities from another large text. Word candidates were removed from a neighborhood if their syntactic classes did not appear in any of the results produced by the Viterbi.

Performance was measured by calculating the average neighborhood size per text word before and after the application of syntax. This statistic is defined as:

$$ANS_t = \frac{1}{N_w} \sum_{i=1}^{N_w} ns_i$$

where N_w is the number of words in the test sample and ns_i is the number of words in the neighborhood for the i^{th} word in the text. The *error rate* is the percentage of words with neighborhoods that do not contain the correct choice after the application of syntax.

4.1. Experimental Design

Experiments were designed to explore several questions about the application of the HMM. The effect of accuracy in neighborhood calculation was determined by applying two alternative models for hypothesis generation. One model produced larger neighborhoods than the other. The effect of using different numbers of parses for a sentence was also explored. In the best case, some number of parses can be found that provide a significant reduction in neighborhood size with a low error rate. The tradeoffs in using first-order versus second-order transition probabilities were also determined. It was expected that better performance would be obtained with second-order probabilities since they would better estimate the effect of language syntax.

4.2. Text Database

A soft copy (ASCII) text sample known as the Brown Corpus was used for the experiments [5]. This text was chosen because it is large (over 1,000,000 words of running text) and every word is tagged with its syntactic class (84 different tags were used in the experiments). The corpus is divided into 15 subject categories or genres that span a range from newspaper reportage to belles lettres. There are 500 individual samples of running text in the corpus and each one contains approximately 2000 words. The number of samples in each genre differs depending on the amount published in that area at the time the corpus was compiled.

4.3. Hypothesis Generation Algorithm

The operation of the hypothesis generation algorithm was simulated by calculating the feature description for a word from pre-defined features for the letters in the word. All the words in a dictionary with the same feature description were used as its neighborhood. Two feature descriptions that produce different neighborhoods were used to demonstrate the effect of neighborhood size on performance.

The feature descriptions are specialized for lower case characters because the experimentation is restricted to text written in lower case. The first feature description includes vertical bars of different heights, dots, and empty spaces. These features were chosen because they can be reliably computed from images of text even if the characters touch one another [3]. When a feature description is applied to a word it yields a symbolic representation that would correspond to the sequence of occurrence of the features in an image of the word. Thus, both "me" and "may" have the same symbolic representation and the neighborhood of "me" is {"me", "may"}.

The second feature set includes all the features in the first description plus the holes produced by topological containment. The addition of this feature provides a finer discrimination in neighborhood calculation, i.e., smaller neighborhoods.

4.4. Experimental Results

The HMM was applied to correct the simulated text recognition results produced by the two models for hypothesis generation described above. Five parses for each sentence in the test samples were output by the HMM and used to filter the neighborhoods. One test sample was randomly selected from each genre of the Brown Corpus. In each case, both first and second-order syntactic class transition probabilities were estimated from the remainder of the corpus.

The original values for ANS_t, before running the HMM, were 36.0 for the first feature description and 4.7 for the second description.

The results of applying the HMM to sample A06 are shown in Table 1. The results on the other samples are very similar. The effect of using up to five different parses is shown by the columns numbered 1 to 5 at the top of the table. The top half of the table refers to the first feature description and the bottom half to the second. Each half also includes an indication of the two orders of transition probability that were used.

The results show a number of interesting effects. It is seen that the first feature description produces the greatest reduction in ANS_t. Values between 80 and 85 percent are typical. However, error rates of between 9 and 14 percent occurred. The second feature description provided more encouraging results. A 65 to 70 percent reduction in ANS_t was obtained with an error rate of between one and two percent.

The modification to the Viterbi algorithm that located a number of alternative parses was also very successful at reducing the error rate with a negligible loss in reduction of ANS_t. The effect of using first-order versus second-order transition probabilities was almost negligible. The results show that the percentage reduction is virtually unaffected by the difference in syntactic information.

5. Discussion and Conclusions

A hidden Markov model for incorporating language-level syntactic constraints in a text recognition

		parses									
feature	trans	1		2		3		4		5	
desc.	order	ErrRate	%reduce	ErrRate	%reduce	ErrRate	%reduce	ErrRate	%reduce	ErrRate	%reduce
1	1	12.82	84.64	10.95	83.07	9.94	81.91	9.43	81.19	8.83	80.30
1	1	12.71	84.66	11.00	83.19	9.94	81.99	9.43	81.29	8.93	80.39
1	1	12.87	84.67	11.00	83.19	9.94	81.99	9.43	81.28	8.93	80.40
1	1	12.82	84.61	10.90	83.02	9.89	81.99	9.28	81.40	8.78	80.20
1	1	12.87	84.56	10.85	83.01	9.84	81.80	9.23	81.02	8.78	80.07
1	1	12.87	84.56	10.85	83.02	9.84	81.86	9.28	81.09	8.78	80.06
1	1	12.87	84.57	11.00	83.08	10.04	81.80	9.38	81.01	8.73	80.19
1	2	12.61	84.77	11.25	83.37	10.09	81.90	9.33	81.03	8.83	80.19
1	2	13.52	84.63	12.01	83.03	10.80	81.74	10.24	81.42	9.23	80.46
1	2	13.02	84.82	11.71	83.57	10.34	81.95	9.43	81.22	8.68	80.37
1	2	12.71	84.77	11.30	83.28	9.94	81.71	9.08	81.34	8.73	80.71
1	2	13.02	84.52	11.60	83.15	10.34	81.64	9.79	80.90	9.18	80.40
1	2	13.72	84.37	12.21	83.19	10.95	81.59	10.29	80.86	9.33	80.38
1	2	13.82	84.88	12.36	83.43	11.20	81.94	10.54	81.25	9.38	80.65
2	1	2.83	69.24	2.07	68.35	1.66	67.42	1.46	66.77	1.21	65.77
2	1	2.83	69.23	2.02	68.36	1.66	67.39	1.41	66.75	1.21	65.68
2	1	2.83	69.23	2.02	68.31	1.72	67.42	1.46	66.73	1.26	65.79
2	1	2.83	69.23	2.07	68.32	1.72	67.42	1.46	66.71	1.21	65.79
2	1	2.83	69.23	2.02	68.32	1.77	67.42	1.46	66.71	1.26	65.80
2	1	2.88	69.23	2.02	68.38	1.77	67.47	1.46	66.72	1.26	65.85
2	1	2.77	69.23	2.02	68.38	1.77	67.47	1.46	66.72	1.26	65.83
2	2	2.17	69.11	1.66	68.19	1.36	67.34	1.11	66.68	1.01	65.94
2	2	3.68	69.01	2.83	67.88	2.42	66.78	2.27	66.16	2.17	65.43
2	2	2.37	69.07	1.72	68.14	1.31	67.31	1.21	66.67	1.01	66.01
2	2	2.22	69.05	1.72	68.24	1.21	67.43	1.16	66.96	1.01	66.10
2	2	2.27	69.13	1.77	68.22	1.41	67.38	1.31	66.85	1.21	66.10
2	2	2.52	69.13	1.87	68.22	1.41	67.38	1.26	66.81	1.11	66.23
2	2	2.42	69.07	1.87	68.23	1.51	67.65	1.26	66.99	1.16	66.39

Table 1. Results of applying the HMM to sample A06.

algorithm was presented and an experimental investigation was performed. It was shown that small initial word groups from the text recognition algorithm, on the order of five choices per word, produced the best performance. Also, the proposed modification of the Viterbi algorithm was quite valuable in reducing the error rate while also providing a reduction in the average neighborhood size of about 65 percent.

It was interesting that there was almost no difference in performance depending on whether first or second-order transitions were used. This is a valuable result since it suggests that a working implementation can be achieved with a small cost in storage.

Acknowledgment

An-Tzu Chin implemented the approach and performed the experiments.

References

1. G. D. Forney, "The Viterbi algorithm," *Proceedings of the IEEE 61*, 3 (March, 1973), 268-278.

2. J. J. Hull, S. N. Srihari and R. Choudhari, "An integrated algorithm for text recognition: comparison with a cascaded algorithm," *IEEE Transactions on Pattern Analysis and Machine Intelligence PAMI-5*, 4 (July, 1983), 384-395.

3. J. J. Hull, "Hypothesis generation in a computational model for visual word recognition," *IEEE Expert 1*, 3 (Fall, 1986), 63-70.

4. J. J. Hull, "Feature selection and language syntax in text recognition," in *From Pixels to Features*, J. C. Simon (editor), North Holland, 1989, 249-260.

5. H. Kucera and W. N. Francis, *Computational analysis of present-day American English*, Brown University Press, Providence, Rhode Island, 1967.

6. R. Kuhn, "Speech recognition and the frequency of recently used words: A modified Markov model for natural language," *Proceedings of the 12th International Conference on Computational Linguistics*, Budapest, Hungary, August 22-27, 1988, 348-350.

7. L. R. Rabiner and B. H. Huang, "An introduction to hidden Markov model," *ASSP Magazine 3*, 1 (1986), 4-16.