Combining Statistical and Syntactic Methods in Recognizing Handwritten Sentences

Rohini K. Srihari and Charlotte M. Baltus
Center for Document Analysis and Recognition
State University of New York at Buffalo
Buffalo, New York 14260 USA
e-mail: rohini@cs.buffalo.edu

Abstract

The output of handwritten word recognizers tends to be very noisy due to factors such as variable handwriting styles, distortions in the image data, etc. In order to compensate for this behaviour, several choices of the word recognizer are initially considered but eventually reduced to a single choice based on constraints posed by the particular domain. In the case of handwritten sentence/phrase recognition, linguistic constraints may be applied in order to improve the results of the word recognizer. Linguistic constraints can be applied as (i) a purely post-processing operation or (ii) in a feedback loop to the word recognizer. This paper discusses two statistical methods of applying syntactic constraints to the output of a handwritten word recognizer on input consisting of sentences/phrases. Both methods are based on syntactic categories (tags) associated with words. The first is a purely statistical method, the second is a hybrid method which combines higher-level syntactic information (hypertags) with statistical information regarding transitions between hypertags. We show the utility of both these approaches in the problem of handwritten sentence/phrase recognition.

Introduction

Recently, there has been a great deal of interest in handwriting recognition systems. Such systems are referred to as on-line or off-line systems. In the former, a light pen and a special tablet are used by the writer to enter phrases or sentences. Such a method of data entry is sometimes more "natural" than using a keyboard. In the latter, ordinary handwriting on paper is scanned and digitized. Since handwriting is highly variable, it is advantageous to use linguistic constraints to derive the correct interpretation.

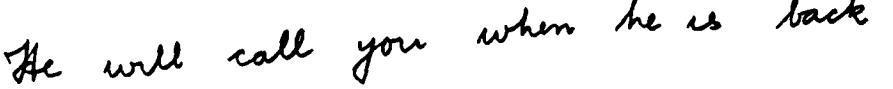
This paper deals with the problem of using linguistic constraints to correct the output of handwritten word recognizers (HWR), specifically when the input consists of cursive handwritten words forming sentences. A handwritten word recognizer (HWR)is employed which, for each input word in the sentence, returns the best n words in the lexicon which match the structural features of that input word. In many cases, the word recognizer often does not return the correct word as its top choice.

As an example, consider Fig. 1 which shows the digitized image of the sentence "He will call you when he is back" along with the output of the HWR. If we restrict the recognizer to return only the top choice for each input word, the sentence will be erroneously read as "He with call pen when he us back". Although the correct words can be found in the top 2 choices, it requires the use of contextual constraints in order to override the (sometimes erroneous) top choice. This example also illustrates the tendency of word recognizers to misread short (less than or equal to 3 characters) words more frequently than longer words. Furthermore, short words tend to be pronouns, prepositions and determiners causing frequent word confusion between very different syntactic categories (e.g., as, an).

In such cases, linguistic constraints may be used to select the best word choice at every word position. The HWR problem is further compounded by the possible absence of the correct word in the top n choices. Even when the actual word is missing, it is often the case that a word having the same syntactic category as the correct word is present. Determining the correct syntactic category of words, is beneficial since it can provide valuable feedback to the HWR. More precisely, it may enable the location of HWR errors (based on highly improbable transitions between syntactic categories and poor HWR scores). Such a technique is cohesive with our philosophy of incorporating language models into the word recognition process, rather than simply using language models for post-processing.

Existing methods to apply linguistic knowledge to this problem consist of purely syntactic methods or purely statistical methods. Syntactic methods employ a grammar capturing the syntax of all possible input sentences and reject those sentence possibilities which are not accepted by the grammar. The problems with such a method are (i) the inability of the grammar to cover all possibilities (especially since informal language is frequently ungrammatical) and (ii) the computational expense involved in parsing.

In this paper, we focus on two statistical meth-



he	with	call	pen	when	he	us	back
she	will	will	you	were	be	is	bank

Figure 1: Digitized image of sentence "He will call you when he is back".

ods having the same objective: to increase the percentage of correctly recognized words/categories (considering only top-choice). Statistical methods ([Keenan *et al.*, 1991, Evett *et al.*, 1991, Hull, 1992]) are forumulated as some variation of Markov models and are based on transition frequencies between the syntactic categories represented by pairs (or n-tuples) of entities in the candidate sentence. In the first method, the entities correspond to individual words and first and second-order transitions between syntactic categories (tags) of words are employed. The problem with this purely statistical approach is that local minima/maxima often cause the correct word/tag choice to be overlooked. As an example, subject verb agreement becomes a problem due to intervening prepositional phrases as in "The folder with all my new *ideas is* missing". The second method discussed in this paper takes a hybrid approach, where both syntactic and statistical knowledge are employed in arriving at the best word/tag choices.

Word Recognition

Since the training phase requires the processing of several thousand sentences, the long and computationally expensive procedure of digitizing followed by recognition is avoided by employing a program which simulates the output of an actual HWR. The actual HWR is implemented as a two-stage procedure. In the first stage, the word-image is segmented into several components; physical features of each component lead to a set of character choices for each segment thus resulting in a set of candidate words. All candidate words which are in the lexicon are returned as the *direct recognition* output of the HWR. In case none of the words are found in the lexicon (a frequent occurence), the candidate words are matched against the lexicon and the top n choices (based on string-edit distances) are output. The latter case is referred to as *post-processing*.

Based on the intermediate results of the actual word recognizer, we have computed statistics which model the behaviour of the first stage. These include substitution, splitting and merging statistics. Given an input word, and the above statistics, candidate (corrupted) words are generated based on simulating and propogating each of the above three types of errors at each character position. Subsequent stages are the same as in the actual HWR. Although the HWR simulator does return confidences, we only use the relative ranking between word choices rather than the absolute confidences. Based on a comparison between the performance of the actual HWR on a given set as well as the simulated performance, the simulator has proven to be a reliable tool in the development stage of our research.

It is important to note that the output of the HWR (and the simulator) degrades when the size of the lexicon is increased. Thus, information restricting the size of the lexicon (based on physical and syntactic features of a word) can be helpful in improving the output of the HWR.

Model for Handwritten Sentence/Phrase Recognition

Figure 2 illustrates a computational model for a handwritten sentence/phrase recognizer. The guiding principle reflected in this model is the necessity of including language models at a very early stage in the decoding (recognition) process. The "tokenization" step is where lines (obtained from a line buffer) in the input image are segmented into words and punctuation thus producing a stream of word/punctuation images for the next stage[1]. It is necessary to incorporate some language information even at this preliminary stage, e.g., splitting up words containing punctuation such as contractions. Output from this stage is placed in a word buffer.

The direct recognition stage of the HWR attempts to recognize words (obtained from the word buffer) based on intrinsic image features alone. It is important to note that this is a bottom-up approach and does not incorporate a lexicon. Some limited linguistic information may be utilized at this stage, for example, if the previous word image contained a '.', the next word image may start with a capital letter. Providing such information to the OCR module increases the reliabil-

[1] In the case of online recognition, such a buffer may not be necessary.

ity of the direct recognition phase. The output of this stage is referred to as a word candidate.

In the next stage, we look for "islands", or tokens that have been directly recognized with a high degree of confidence. If for example, the initial character of the token was a digit, we could restrict the lexicon used in the post-processing phase of HWR to those words that began with a digit (e.g., 2nd). The next stage uses the (possibly reduced) lexicons in order to find the "closest"[2] actual words to the word candidate. The output is a word neighbourhood; presently we are using a fixed size (3) for the neighbourhood. As the bottom-up recognition improves, we plan on selecting word neighbourhoods based on the confidences provided by the first stage.

The next stage, which uses semantic filtering to reduce the word neigbourhoods is beyond the scope of this paper and will not be discussed here. It is the last stage, syntactic processing, which is the focus of this paper. The input to this stage consists of a sentence/phrase buffer containing the word neighbourhoods for each word. The output of this stage is either (i) a final word choice for each word in the sentence/phrase, (ii) feedback information to the HWR post-processor in terms of the most probable sequence of syntactic categories thus enabling restricted lexicons to be utilised, or (iii) feedback to the direct recognition stage of the HWR, e.g., if syntactic analysis has determined that a particular token must be alphabetic only (as opposed to a mixed alphanumeric string generated by the HWR), this information could be efficiently incorporated into the OCR in a second "reading" of the word image.

Statistical Analysis of Syntax

As stated earlier, current Handwritten Word Recognizers tend to produce output that is highly error prone. Further, the broader the application area, the worse the performance of the HWR post-process, as the lexicon must be expanded to include a broader range of possibilities. The number of errors is significantly lower, however, if one includes the top several choices of the word-level HWR, rather than singling out only the top choice. In order to utilize this information, we must incluce wider contextual information in selecting from among the choices for any given position in the sentence.

We have been able to improve the performance of a HWR system by incorporating statistical information at the word sequence level. The performance improvement derives from selection of lower-rank words from the HWR output when the surrounding context indicates such selection makes the entire sentence more probable. Ultimately, however, this level of analysis must provide feedback to the HWR postprocess so that errors of omission from the original HWR output can

[2] using a string-edit distance

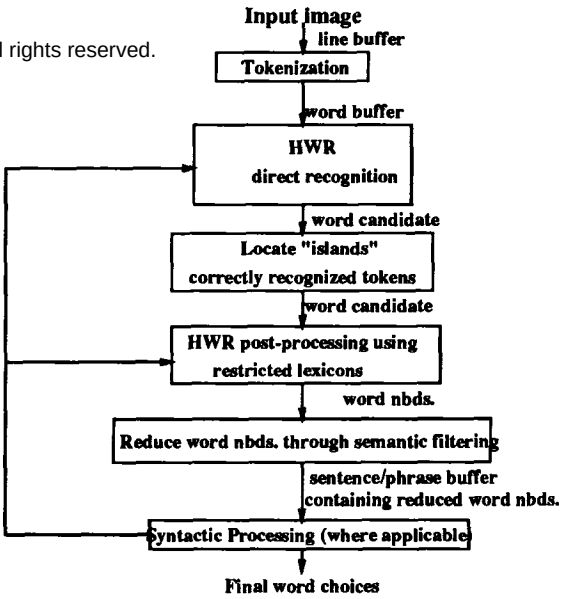


Figure 2: Overall Model of Sentence/Phrase Recognition

be corrected. A method for detecting typographical errors has been discussed in [Garside *et al.*, 1987].

For example, using a 21,000 word lexicon derived from the tagged Wall Street Journal Corpus, HWR simulation on a portion of the WSJ Corpus produced the correct word as the top choice 58% of the time, while the correct word was among the top three choices 69% of the time. Use of syntax to select among the candidates could produce, at best, a 69% correct output. The system becomes practical only if we can accurately pinpoint the errors, allowing use of feedback to the HWR post-process. If such feedback included a reduced lexicon in each iteration, then with each iteration the string-matching would be increasingly likely to produce the correct word.

As a first step in achieving this ultimate objective, we have concentrated on improving the HWR post-process output by selecting second and third choice words when indicated by the surrounding context. For this purpose, a purely statistical model of syntax has been implemented. The course of the experiment covers roughly two phases – a training phase, during which the statistics are generated, and a testing phase, during which the statisitics are used to select from among HWR output word choices. The resulting selected sentences are then compared against the actual input sentences to measure the success of the method. This model is patterned after that of [Hull, 1992] and [Keenan *et al.*, 1991], with the notable difference that in our model, the correct word may not be present at all, and is in fact missing in some 30% of the cases. The fact that our system is capable of recovering from these errors is an indication of its strength. It remains

to be seen whether we can pinpoint those words which are suspect in order to iteratively improve the HWR performance.

The Training Set

Our intent is to use an informal language genre, since most handwritten text falls into this category. More specifically, we will be using collections of electronic mail messages as our training and test sets, since they are readily available and reflect the type of language we are modeling. Such a training set has been collected and tagged using Church's tagger with statisitcs derived from the tagged Brown Corpus. Problems with incorrectly tokenized and tagged words are impeding our efforts to the extent that we have found it necessary to use an alternate (manually corrected) training set to validate our theories. A simultaneous effort is underway to properly tag the e-mail corpus. For the time being, we are using the Wall Street Journal Corpus [3] in which each word has been tagged to indicate its syntactic category. These categories correspond roughly to parts of speech, and most words can belong to one of several categories, depending on context. The particular tagset used should be chosen so that likely transitions among the categories are distinguishable, while at the same time, useful transition probabilities are not lost by use of too broad a category. We selected the tagset used in the tagged WSJ corpus containing 47 categories. This tagset had been derived for use in part-of-speech tagging, an area in which the function of the tagset is analogous to ours. Unlike our electronic mail collection, the WSJ data had been carefully examined for errors, so we were far more confident of the statistics we obtained. It is hoped we can resume the analysis of the e-mail data at some point in the future.

Two related lexicons are needed in our system – one is used for the HWR post-process string matching, while the other is used to retrieve the tag candidates and probabilites for each word generated by the string match (see figure 4). Since our HWR simulator does not currently recognize upper case input, all test and training data was converted to exclusively lower case prior to training and testing. For expediency, the lexicon used for string-matching was generated from only a portion of the WSJ corpus, but which included the test set so that the possibility existed of finding the correct word in each case. The tag-specific lexicon, on the other hand, was derived from the entire corpus, including the test set, so that the correct usage of the word (if the correct word was present) would be included, and the word probability statistics would be as accurate as possible.

Transition statistics were collected over the entire corpus excluding the test set, enabling us to determine whether the system is capable of accurately selecting sequences which have not specifically been seen before.

The Analysis Model

For the remainder of this section, the notation $A : B$ indicates the case where word A has been assigned the tag B. For each sentence analyzed, we form a word:tag lattice representing all possible sentences for the set of word choices output by string matching (see figure 3). [4]

Any path through the lattice is then evaluated as the product of the transition probabilites among the nodes and the word probabilites at each node.

Transition probabilities describe the likelihood of tag following some preceding (sequence of) tag(s). One question is whether and to what degree the length of the transitions effects the results. Use of non-local information can introduce meaningless information and involves a high overhead in storage, while excessive restriction on transition length may eliminate useful information. Two versions of the syntax model have been implemented – one uses first order transition statisitcs, while the other uses second order statistics. Little performance change was noted between the two [5], so we have concentrated on the first order model for the time being. These statistics are calculated during training as:

$$P(tag_B|tag_A) = \frac{\#(tag_A \rightarrow tag_B)}{\#(tag_A)}$$

Beginning- and end- of-sentence markers are incorporated as tags themselves to obtain necessary sentence-level information.

For the word probabilites at each node in the lattice, several pieces of information are incorporated: [6]

- *grammatical frequency*, or the conditional probability $P(Tag|Word)$, calculated as:

$$P(Tag|Word) = \frac{\#(Word : Tag)}{\#(Word : AnyTag)}$$

  Rarity tags [Garside *et al.*, 1987] may be used for efficiency.

- *word frequency*, which is defined as:

$$\frac{\#(Word : AnyTag)}{\#(AnyWord : AnyTag)}$$

- tag frequency, calculated as:

$$tag\ frequency = \frac{\#(AnyWord : Tag)}{\#(AnyWord : AnyTag)}$$

[3] a collection of 2.5 million words obtained from articles from the Wall Street Journal, as collected and tagged by the UPENN Treebank Project

[4] the presence of the DT tag in the trellis is explained below

[5] this result agrees with the findings of [Keenan *et al.*, 1991]

[6] eventually, we expect to incorporate a weighting factor based on the HWR ranking

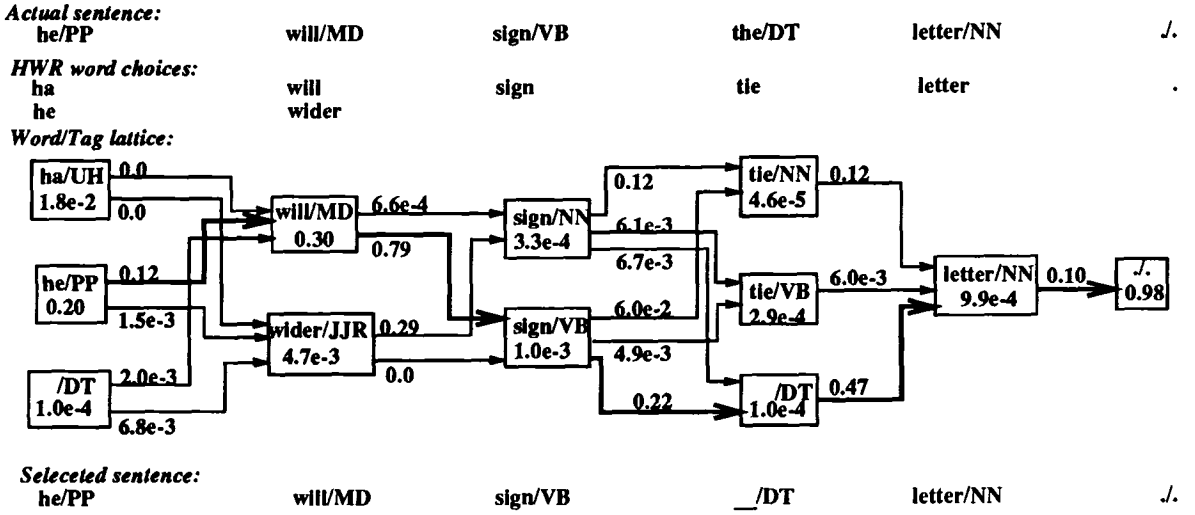


Figure 3: Sample Word:Tag Lattice For Analysis of HWR choices

The product of the first two factors is normalized by dividing by the tag frequency to obtain the word-probability:

$$word\,probability = \frac{grammatical\ frequency * word\ frequency}{tag\ frequency}$$

$$= \frac{\#(Word:Tag)}{\#(AnyWord:Tag)}$$

$$= P(Word|Tag)$$

A Viterbi algorithm is used to find the top N Word:Tag sequences through the lattice. At present, only the top sentence candidate is being evaluated.

Testing Phase

Figure 4 shows the word recognition test system. The test set contained 93 sentences (approximately 2,000 words) from the WSJ corpus. Evaluation of test results excluded any words known to be 100% correctly 'recognized' by our HWR simulator (e.g. words containing punctuation). Three statistics are reported. The percent of words for which the correct tag is present in the lattice indicates the best our system can achieve under its present configuration. The word:tag sequence having the highest score is compared to the (correctly-tagged) actual input sentence. The percent of correct tags and percent of correct words in the derived sentence are then calculated.

Results

Our first run resulted in only a slight improvement over the accuracy of the HWR output (see Table 1). We noted that minute errors in the tagging of the training set were being introduced into the word:tag lattice as word:tag possibilities where none should exist. To eliminate erroneous entries, we eliminated from the lattice any candidate word:tags having a grammatical frequency below the threshold value of 0.3. The results (displayed in Run 2 in Table 1) indicated a further improvement in accuracy in the top choice sentence, even though we reduced the percentage of correct tags occurring in the lattice overall. The chosen path is illustrated in boldface. Further analysis showed that the correct tag most frequently missing from the lattice was the DT (determiner) tag, which was consistent with the fact that (i) the shorter words are more frequently missed by the HWR, and (ii) words of the category DT tend to be short words. We utilized this information by including the DT tag in the lattice in all cases of short words (< 4 characters) where the DT tag was not otherwise a candidate (it was also necessary to assign a default word probability to the node). The lattice of Figure 3 demonstrates this procedure being used to derive the correct tag sequence even when the correct word ('the') was not output by the HWR. The results, shown as Run 3 in the Table, show that this type of fine-tuning can further improve accuracy in tag selection. The fact that the accuracy of the word sequence decreased is not problematic, since these are presumably cases in which the DT tag was selected with no associated word. In such cases, feedback to

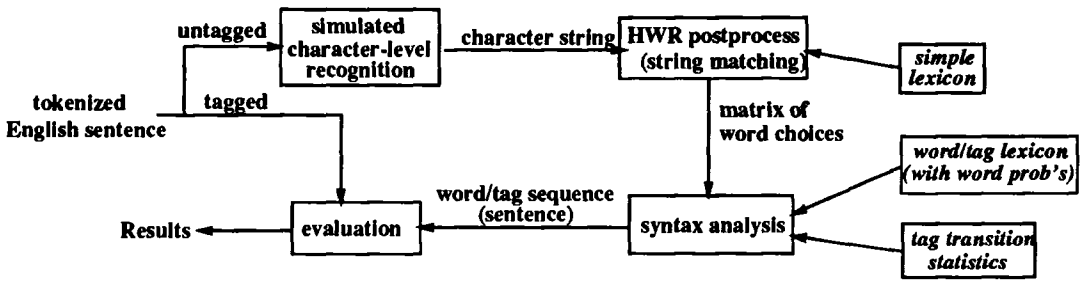


Figure 4: Test Phase – Statistical Analysis of HWR Output

the HWR with a lexicon containing only words of the class DT would be highly accurate.

HWR simulation:
% correct of topchoice words = 58.0
% correct word present = 69.1

	% correct tag present	% correct tags in top sequence	% correct wds in top sequence
Run 1: GF threshold: none	77.1	60.6	59.7
Run 2: GF threshold: 0.3	71.3	63.2	61.1
Run 3: GF threshold: 0.3 DT tag added for short words	74.8	64.2	60.7

Table 1: Test Results – Statistical Syntax Model

Hybrid Method

We attempt to overcome the problems associated with the first approach by employing a hybrid method, one which combines both syntactic as well as statistical knowledge. As in the case of the earlier method, this is also broken into training and testing phases. The training phase has been completed, the testing phase is under progress.

Training

The first stage, known as the *training stage*, consists of partitioning sentences into meaningful syntactic units (*hypertags*[7]) based on the part-of-speech tag that is associated with each word. Some of the (40) hypertag types loosely correspond to noun, verb, prepositional, adjective and adverbial phrases. For example the sentence "The new leader of our group will be presenting it" will be partitioned as follows:

[7]This notation was taken from [Garside *et al.*, 1987].

[NP-DET-SING (The DET) (new JJ) (leader NN)]
[PP-NP (of IN) (our PP$) (group NN)]
[VP-VBG-3 (will MD) (be BE) (presenting VBG) (it PP3)]

The bracketing indicates the scope of hypertags where the first entry is the hypertag type (e.g., NP-DET-SING indicates a noun hypertag containing a determiner and a singular noun). The entries following each word represent the tag assigned by the part of speech tagger. A hypertag may consist of a single word (e.g., conjunctions). Verb hypertags are classified based on (i) basic tense, and (ii) the presence of subject and object pronouns. Noun hypertags are classified based on (i) number (singular, plural, ambiguous) and (ii) the presence of articles or determiners. Due to this classification scheme, there are several types of noun hypertags, and even more types of verb hypertags. The complexity of parsing is eliminated since a regular expression recognizer is sufficient for grouping the words into hypertags.

This stage is strongly motivated by the idea of "parsing by chunks" [Abney, 1991a, Abney, 1991b] and the theory of *phi-phrases* [Gee and Grosjean, 1983]. Due to the problems with tag n-grams described above, a method of naturally partitioning sentences into hypertags was desired. Phi-phrases correspond to prosodic phrases in sentences and thus represent a "natural" partitioning of sentences. Chunks on the other hand represent mid-level sentence constituents and are based on syntactic principles. [Abney, 1991a] shows that phi-phrases are formed by "sweeping" orphaned words (such as subject and object pronouns) into neighbouring chunks. Furthermore, a precise syntax for chunks is given. Finally, the theory states that the syntax of structures is rigid; it is the relationship between chunks (i.e., the structure of sentences) which is flexible. This last statement has been the motivation for the hybrid approach being described here.

The idea of "hypertags" differs in some key ways from chunks and phi-phrases. Hypertags in our system are a combination of chunks and phi-phrases. This

combination has been designed with sentence recognition (rather than sentence understanding) as the goal. Phi-phrases are used to group subject and object pronouns with verb hypertags (e.g., I told him). We also allow for possessives within PP hypertags (e.g., in John's house). However, constituents such as Wh-NP's (e.g., that) are not grouped with neighbouring chunks. The inclusion of these into neighbouring hypertags does not enable further syntactic checking within that hypertag. Furthermore, they are part of the overall sentence structure (which is represented by the statistical data regarding transitions between hypertags). Finally, hypertags provide a natural partitioning of the sentence in order to apply semantic knowledge. The head words of each hypertag can be used in statistics involving collocation and dictionary definition overlap.

Based on the sentences that have been processed by the above method, we compute the following statistics:

- transition frequencies $P(H_B|H_A)$ defined as

$$\frac{no.\ of\ transitions\ from\ H_A\ to\ H_B}{no.\ of\ transitions\ from\ H_A}$$

  where H_A and H_B are hypertags.

- hypertag probabilities

$$P(TS_i|H_j)$$

  over all possible hypertags H_j, all possible observed tag sequences TS_i

Testing

As in the previous method, here also we have a lattice of possibilites which must be traversed in an optimal manner. In this case, the entries correspond to word sequence/hypertag pairs rather than word/tag pairs. We are experimenting with a dynamic method (one that avoids enumeration of all possible sentences) of finding the sequence of hypertag transitions $\bar{Z}$ which maximizes the conditional probability $P(\bar{Z}|\bar{X})$ where $\bar{X}$ is the observed word sequence. Central to the technique is an adaptive-length Viterbi procedure. Initial results have been promising.

Summary

A model for sentence/phrase recognition has been outlined with emphasis on improving the noisy output of the handwritten word recognizer, the central component of the system. We have presented two statistical methods of incorporating syntactic constraints into the system. A noticeable improvement has been demonstrated by using the first technique. The hybrid method which incorporates both syntactic as well as statistical knowledge is currently in the testing stage.

References

Abney, Steven P. 1991a. Parsing by Chunks. In Berwick, Robert C.; Abney, Steven P.; and Tenny, Carol, editors 1991a, *Principle-Based Parsing: Computation and Psycholinguistics.* Kluwer Academic Publishers, Boston. 257–278.

Abney, Steven P. 1991b. Rapid Incremental Parsing with Repair. to appear.

Evett, L. J.; Wells, C.J.; Keenan, F.G.; Rose, T.; and Whitrow, R.J. 1991. Using Linguistic Information to Aid Handwriting Recognition. In *Proceedings of the International Workshop on Frontiers in Handwriting Recognition.* 303–311.

Garside, Roger; Leech, Geoffrey; and Sampson, Geoffrey 1987. *The Computational Analysis of English.* Longman.

Gee, J. and Grosjean, F. 1983. Performance Structures: a Psycholinguistic and Linguistic Appraisal. *Cognitive Psychology* 15:411–458.

Hull, Jonathan J. 1992. Incorporation of a Markov Model of Language Syntax in a Text Recognition Algorithm. In *Proceedings of Symposium on Document Analysis and Information Retrieval.* 174–185.

Keenan, F.G.; Evett, L.J.; and Whitrow, R.J. 1991. A large vocabulary stochastic syntax analyser for handwriting recognition. In *Proceedings of the First International Conference on Document Analysis (ICDAR-91).* 794–802.