TOWARD ISLAND-OF-RELIABILITY-DRIVEN VERY-LARGE-VOCABULARY ON-LINE HANDWRITING RECOGNITION USING CHARACTER CONFIDENCE SCORING

John F. Pitrelli, Jayashree Subrahmonia, and Benoît Maison

IBM T. J. Watson Research Center
P. O. Box 218, Yorktown Heights, NY 10598 U.S.A.
{pitrelli,jays,bmaison}@us.ibm.com

ABSTRACT

We explore a novel approach for handwriting recognition tasks whose intrinsic vocabularies are too large to be applied directly as constraints during recognition. Our approach makes use of vocabulary constraints, and addresses the issue that some parts of words may be written more recognizably than others. An initial pass is made with an HMM recognizer, without vocabulary constraints, generating a lattice of character-hypothesis arcs representing likely segmentations of the handwriting signal. Arc confidence scores are computed using *a posteriori* probabilities. The most-confidently-recognized characters are used to filter the overall vocabulary, generating a word subset manageable for constraining a second recognition pass. With a vocabulary of 273,000 words, we can limit to 50,000 words in the second pass and eliminate 39.3% of the word errors made by a one-pass recognizer without vocabulary constraints, and 18.3% of errors made using a fixed 30,000-word set.

1. INTRODUCTION

Applying vocabulary constraints significantly aids handwriting recognition accuracy [8] by providing a substantial reduction in character-level perplexity compared with allowing any character to follow any other character. However, the drawback is that words not in the vocabulary cannot be recognized. For very-large-vocabulary applications, such as those involving proper names, the vocabulary may be unmanageably large for the recognizer, due to speed, memory, or other implementation limitations, impeding exploitation of vocabulary constraints.

A second problem is that different parts of words may be written more recognizably than others. In the case of typical left-to-right algorithms for on-line recognition, such as HMM systems, if a poorly-written character occurs early in the word, it may set the search astray, making it impossible to recover the correct answer. This is particularly likely in the case of a larger-vocabulary system, whose search algorithm requires substantial pruning of hypothesized transcripts of the beginning portion of the word.

An appealing method for dealing with this latter problem is an "island-of-reliability" approach, in which reliably-recognized characters are used as anchor points from which to constrain and expand hypotheses. While such systems do exist [7], research in this direction is hampered by the complexity of formulating search strategies to contend with partial recognition hypotheses with varied patterns of labeled and unlabeled segments spanning the handwriting signal.

We explore a hybrid approach to address both problems. A first, "character-set", recognition pass is performed without employing vocabulary constraints. *A posteriori* confidence scoring [4] is performed on the character arcs in the hypothesis lattice, leading to identification of the most-confidently-recognized characters. Words conforming to this character pattern are extracted from a very large vocabulary, yielding a word set small enough to be usable to constrain a second, "word-set", recognition pass.

In general, we will use the term "vocabulary" to refer to the set of known words intrinsic to an application, which we assume to be too large to apply directly as a recognition constraint. A vocabulary subset which is used to constrain recognition will be referred to as a "word set".

2. CHARACTER CONFIDENCE SCORING

Our base system is a hidden-Markov-model-based (HMM) recognizer which seeks to label any handwriting signal with the highest-*a-posteriori*-probability word sequence:

$$\hat{c} = \underset{c \in \mathcal{C}}{\operatorname{argmax}} \; P(c|o) \tag{1}$$

where o denotes the observed signal, c denotes a character or word sequence, $\mathcal{C}$ denotes the set of all possible such sequences, and $\hat{c}$ denotes the highest-probability sequence. $P(c|o)$ is decomposed as follows:

$$P(c|o) = P(o|c)P(c)/P(o) \tag{2}$$

$P(o|c)$ is a likelihood computed using a "character-shape model", a statistical model of *how* people write a given text. This model is typically applied after segmenting o into frames defined by some windowing criterion. $P(c)$ comes from a statistical language model of *what* people are likely to write; it can be in terms of characters (*e. g.* a character N-gram) or words (*e. g.* a word unigram). Typically, $P(o)$ is disregarded because it is constant relative to c, so $P(o|c)P(c)$ constitutes the score for hypothesis c.

For *a posteriori* character scoring, however, we must compare $P(c|o)$ for characters spanning different sequences of frames of the handwriting signal, and so $P(o)$ cannot be neglected. We decompose $P(o)$ into $\sum_c P(o|c)P(c)$. We recognize $P(o|c)P(c)$ as the score we use during typical recognition, and so in effect we are normalizing each score against the sum of all character scores for a given segment of the signal. We approximate the set of all scores using an N-best-words list, analogously to Stolcke *et al.* [9] except we do it at the character level rather than at the word level.

Scoring proceeds as follows. The set of character hypotheses output by the recognizer is represented as a directed graph with one start node and one end node. Each arc carries a character along with its likelihood score, and a start and end node, with each node linked to a frame index indicating its position in the handwriting signal. Each path traversing the graph defines one word hypothesis, whose likelihood is computed as the product of its arc likelihoods, assuming independence. Likelihoods are scaled by raising to a power α between 0 and 1. Setting $\alpha < 1$ reduces the probabilities of the top paths, which can be interpreted as compensating for (1) pruned paths, (2) uncertainty about the accuracy of the character-shape model, and/or (3) multiplication of probabilities which were possibly not fully independent. In these experiments, α is empirically set to 0.2.

Path probabilities are computed by dividing each path likelihood by the sum of all path likelihoods. Each *arc posterior* is then computed as the sum of the probabilities of all paths that include that arc. Next, we compute a confidence score for each character for each frame, which we define to be the sum of the posteriors of all arcs which span that frame and carry that character. Finally, for each arc, we compute *arc confidence* to be the average over all the arc's frames of the arc's character frame confidences. More details are provided by Maison and Gopinath [4] and Kemp and Schaaf [3]; see also [1] and [5].

A simple example is shown in Figure 1; we will focus on the longer "d" arc, and let $\alpha = 0.5$ for this example. Path likelihoods for "dog", "day", "clog" and "clay" are 10^{-2}, 2×10^{-3}, 10^{-3}, and 2×10^{-4}, respectively, obtained by multiplying their arc likelihoods and raising to the power α. The path probability for "dog" is its likelihood divided by the sum of all the path likelihoods, $10^{-2}/(10^{-2} + 2 \times 10^{-3} + 10^{-3} + 2 \times 10^{-4}) = 0.758$. This is also the arc posterior for the "d" in "dog", as "dog" is the only word traversing this arc. Next, we compute the character frame confidence score for this arc's frames, which is the sum of the posteriors of all arcs which span that frame and carry that character. For frame 6, this is the only "d" arc, so 0.758 is the "d" frame confidence. For the preceding frames, however, we add the other "d" arc posterior, which is the path likelihood for "day", the only word traversing

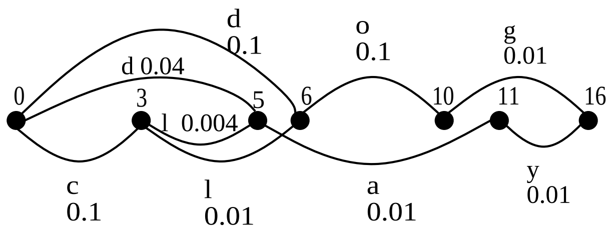

Fig. 1. Example character-hypothesis graph to be used to compute character-arc confidences. This graph represents 16 frames spanned by four hypothesized words, "dog" (likelihood = $0.1 \times 0.1 \times 0.01 = 10^{-4}$), "day" ($4 \times 10^{-6}$), "clog" ($10^{-6}$) and "clay" ($4 \times 10^{-8}$). Nodes are labeled here by the number of frames preceding them.

this arc, $2 \times 10^{-3}/(10^{-2} + 2 \times 10^{-3} + 10^{-3} + 2 \times 10^{-4}) = 0.152$. This yields a "d" frame confidence of 0.909 for these frames. Finally, we compute arc confidence, which is the average of character frame confidence over all its frames. Arc confidence for the longer "d" arc then is the average of 0.909 for five frames and 0.758 for frame 6, or 0.884.

3. RECOGNITION SYSTEM

Experiments are performed using the IBM on-line handwriting recognizer in the IBM Ink Manager$^{\text{TM}}$ software; previous papers [6] [10] describe most algorithms in detail. Data are collected as a stream of (x, y) points indexed in time, re-sampled to be equi-distant. Features based on distances and angles are computed at each point. Windows of temporally-adjacent points are assembled around window centers, which are mainly local extrema in x and y and the strokes' pen-down and pen-up points. Feature vectors of the points within a window are spliced together to form window feature vectors. These are projected onto a lower-dimensional space; the resulting vectors become the frames.

In our system, each character is represented by a set of four allograph HMMs. Each allograph model consists of a sequence of states; sequences vary in length. Mixture-Gaussian models, trained using an EM algorithm, represent the distribution of frame vectors for each state, $P(o|c)$. Beam search, governed by word set or character set, begins with a forward pass using fast-match character-shape models, and a character-level language model $P(c)$ in the case of character-set recognition. Then, hypothesized words are optionally re-scored using a word-unigram language model. Finally, the hypotheses are re-scored using detailed-match character-shape models. Note that each "recognition pass" mentioned elsewhere in this paper consists of all these steps.

All experiments reported here are based on running the system in American English writer-independent mode. Each experiment uses character models trained on an in-house database of approximately 165,000 words plus 330,000 discrete characters provided by 450 writers.

4. DATABASE

Testing is done using the isolated-word portion of the Unipen database [2], Train-R01/V07. We restrict to fully-alphabetic English and proper-name inputs – 26 lower- and 26 upper-case letters. We restrict attention to words which are four or more letters long. The data set was further reduced by a factor of three down to 8009 word tokens.

A 273,000-word unigram and vocabulary were derived from a text corpus of 600 million words drawn from various sources including news and office correspondence. A portion of this corpus was also used to derive a set of 30,000 most common words, for baseline evaluations, and a character-4-gram language model, for character-set recognition.

5. EXPERIMENTAL PROCEDURE

A character-set recognition pass is run using the character 4-gram as the statistical language model. Search beam parameters are set wide, allowing 150 hypotheses per frame in the character-level search lattice, and 70 best word outputs, yielding a large lattice of character arcs to approximate the full set of hypotheses for computing *a posteriori* arc probabilities. Arc confidences are computed for each character in the recognized character sequence, and the M most-confidently-recognized characters are selected as "islands of reliability". We begin with $M = 2$, empirically chosen to obtain word sets smaller than 50,000 words.

A word template is formed from these M characters. The template consists of the characters in the order in which they occur in the hypothesis, with character wild-cards inserted between them if they span non-adjacent frames, before them if neither reaches the first frame, and after them if neither reaches the last frame in the handwriting signal. The word template is then used to filter the vocabulary, yielding a word set with which to constrain the second recognition pass. Because this pass is intended primarily to generate first-choice word results, a narrower search beam is used, 70 hypotheses per frame in the character-level search lattice. In the initial experiments, we do not employ word unigrams.

We measure results against two baseline conditions, both using the narrower beam parameters as appropriate for producing single-hypothesis recognition results. A single-pass "word-set" baseline is constrained using the 30,000-word set but not using the word unigram. A "character-set" baseline consists of a single pass constrained only by character set, using the same character 4-gram language model.

6. RESULTS

Our experimental system yields a net improvement in accuracy over both baseline systems. It eliminates 3.6% of the word-recognition errors produced by the word-set baseline system. This is mainly due to our system enjoying an advantage of being able to access almost ten times as many words as the word-set baseline. In fact, 63% of the errors made by the baseline system are due to words being out of the 30,000 word set. By making the 273,000-word vocabulary available, we prevent 83% of those out-of-word-set errors, or 52% of all the baseline's errors, from being made *a priori* by the word-set constraint, though some errors remain due to mis-recognition. The 3.6% reduction shows that the experimental system's advantage of having the large vocabulary available outweighs its disadvantage of having to commit to two characters of the output before being able to apply a word-set constraint, compared to the baseline system exploiting its word-set constraint from the beginning.

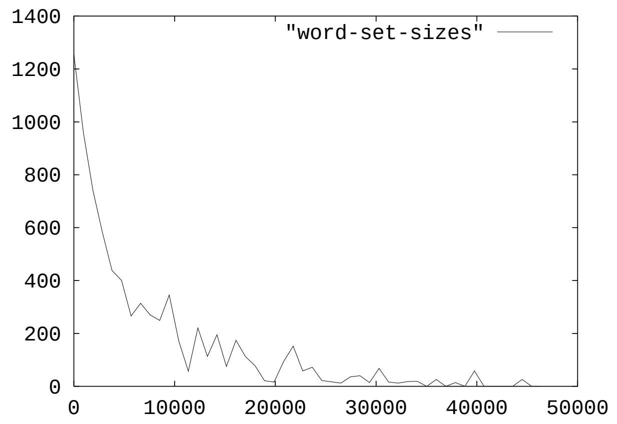

Fig. 2. Histogram of word-set sizes resulting from matching two-character patterns against a 273,000-word vocabulary.

The system eliminates 34.4% of the character-set baseline's word-recognition errors, confirming the benefits of using word-set constraints even when they are not applied until after committing to the recognition of two characters.

Of the errors made by the system, 61% are due to the word not being in the word set, which breaks down to 11% missing from the vocabulary and 50% being excluded by an erroneous two-character pattern. Thus, it is desirable to loosen the character patterns, employing larger word sets to reduce the 50%; next we investigate whether it is practical.

Figure 2 shows a histogram of sizes of word sets resulting from using two-character patterns. We note that by fixing $M = 2$ we narrowly succeed in getting all of the word sets below 50,000 words. However, the average is only 7859 words, indicating it should be possible to modify the algorithm to produce larger word sets. The importance of doing so is underscored by the observation that 3% of the word sets are under 200 words, and a few are actually empty.

Therefore, we repeated this experiment, but setting M separately on each input to be the fewest number of characters which results in a set of 50,000 or fewer words. In fact, 54% of the time, choosing one character reduces the word set sufficiently. As a result, the second recognition pass's average word set expands to 18,268 words. Error reductions improve to 5.7% relative to the word-set baseline, and 35.8% relative to the character-set baseline. Now only 40%

Experimental Condition	% Error reduction relative to: Word-set baseline	Character-set baseline
No word unigram		
$M = 2$	3.6	34.4
variable M	5.7	35.8
With word unigram		
variable M	18.3	39.3

Table 1. Summary of results.

of errors are attributable to an erroneous character pattern excluding the correct word from being found in the vocabulary when assembling the word set for the second pass.

Finally, we repeated this experiment with the addition of a word-unigram language model during the second pass of recognition. Accordingly, we generated new baseline conditions with the same unigram. Further improvement was realized; the experimental condition eliminate 18.3% of the errors made by the word-set/word-unigram baseline, and 39.3% of errors made by the character-set/word-unigram baseline. Results are summarized in Table 1.

7. CONCLUSION

We apply a two-pass algorithm to handwriting recognition, designed to yield some of the benefit of an island-of-reliability approach, while essentially retaining the manageable and well-established search strategy of a left-to-right system such as HMM. Using a first pass unconstrained by word set, character-confidence results can be used to extract a manageable subset of a very large vocabulary, thereby providing word-set constraints to be exploited in a second pass. By manipulating the number of character islands accepted from the first pass, the word-set size can be constrained to one which harmonizes well with the other parameters of the recognizer, such as the pruning criteria in the search.

8. FUTURE WORK

“Very-large-vocabulary” applications are generally actually open-vocabulary tasks. Ideally, recognition should not be ultimately constrained by any finite vocabulary. To this end, future work should include applying confidence measures to the results of the word-set recognition pass. If confidence is low enough, recognition could revert back to a character-set mode, to recognize words outside the vocabulary.

9. ACKNOWLEDGMENTS

We gratefully acknowledge Eugene Ratzlaff and Michael Perrone for assistance with the recognizer and the paper, and Alain Biem for assistance with the paper.

10. REFERENCES

[1] Evermann, G., and P. C. Woodland, “Large Vocabulary Decoding and Confidence Estimation using Word Posterior Probabilities”, *Proceedings of ICASSP 2000: IEEE International Conference on Acoustics, Speech, and Signal Processing*, Istanbul, Turkey, June, 2000, pp. 2366-2369.

[2] Guyon, I., L. Schomaker, R. Plamondon, M. Liberman, and S. Janet, “UNIPEN Project of On-Line Data Exchange and Recognizer Benchmarks”, *Proceedings of the 12th International Conference on Pattern Recognition (ICPR ’94)*, Jerusalem, October, 1994, pp. 29-33.

[3] Kemp, T. and T. Schaaf, Estimating Confidence using Word Lattices, *Proceedings of the Fifth European Conference on Speech Communication and Technology (Eurospeech ’97)*, Rhodes, Greece, September 22-25, 1997, pp. 827-830.

[4] Maison, B., and R. Gopinath, “Robust Confidence Annotation and Rejection for Continuous Speech Recognition”, these proceedings.

[5] Mangu, L., E. Brill, and A. Stolcke, “Finding Consensus among Words: Lattice-Based Word Error Minimization”, *Proceedings of Eurospeech ’99*, Budapest, Hungary, September 5-9, 1999, v. 1, pp. 495-498.

[6] Nathan, K. S., H. S. M. Beigi, J. Subrahmonia, G. J. Clary, and H. Maruyama, “Real-Time On-Line Unconstrained Handwriting Recognition using Statistical Methods”, *Proceedings of ICASSP ’95*, Detroit, Michigan, U. S. A., May 8-12, 1995, v. 4, pp. 2619-2622.

[7] Neskovic, P., and L. N. Cooper, “Neural Network-Based Context Driven Recognition of On-Line Cursive Script”, *Proceedings of the Seventh International Workshop on Frontiers in Handwriting Recognition (IWFHR 7)*, Amsterdam, The Netherlands, September 11-13, 2000, pp. 353-362.

[8] Pitrelli, J. F., and E. H. Ratzlaff, “Quantifying the Contribution of Language Modeling to Writer-Independent On-Line Handwriting Recognition”, *Proceedings of IWFHR 7*, Amsterdam, The Netherlands, September 11-13, 2000, pp. 383-392.

[9] Stolcke, A., Y. König, and M. Weintraub, “Explicit Word Error Minimization in N-Best List Rescoring”, *Proceedings of Eurospeech ’97*, Rhodes, Greece, September 22-25, 1997, v. 1, pp. 163-166.

[10] Subrahmonia, J., K. Nathan, and M. Perrone, “Writer Dependent Recognition of On-Line Unconstrained Handwriting”, *Proceedings of ICASSP ’96*, Atlanta, Georgia, U. S. A., May 7-11, 1996, v. 6, pp. 3478-3481.