ERROR CORRECTION VIA A POST-PROCESSOR FOR CONTINUOUS SPEECH RECOGNITION *

Eric K. Ringger *James F. Allen*
Department of Computer Science; University of Rochester; Rochester, New York 14627-0226
{ringger, james}@cs.rochester.edu

ABSTRACT

This paper presents a new technique for overcoming several types of speech recognition errors by post-processing the output of a continuous speech recognizer. The post-processor output contains fewer errors, thereby making interpretation by higher-level modules, such as a parser, in a speech understanding system more reliable. The primary advantage to the post-processing approach over existing approaches for overcoming SR errors lies in its ability to introduce options that are not available in the SR module's output. This work provides evidence for the claim that a modern continuous speech recognizer can be used successfully in "black-box" fashion for robustly interpreting spontaneous utterances in a dialogue with a human.

1. INTRODUCTION

Existing methods for continuous speech recognition do not perform as well on spontaneous speech as we would hope. Even state of the art recognizers such as Sphinx-II [7] [1] and a recognizer built using HTK [14] [2] achieve less than 60% word accuracy on fluent speech collected from conversations about a specific problem with the TRAINS-95 system [1].

Here are a few examples of the kinds of errors that occur when recognizing spontaneous utterances. They are drawn from problem-solving dialogues that we have collected from users interacting with the TRAINS-95 system. [3] Some errors are simple one-for-one replacements, such as this one:

```
REF:  RIGHT SEND THE  TRAIN FROM MONTREAL TO CHARLESTON
HYP:  RATE  SEND THAT TRAIN FROM MONTREAL TO CHARLESTON
```

Here is an utterance with a replacement of a single word by multiple smaller words:

```
REF:  GO FROM CHICAGO TO TOLEDO
HYP:  GO FROM CHICAGO TO TO LEAVE AT
```

The following utterance contains a more complex example in which adjacent words are misrecognized and in which the hypothesized words overlap the boundary between the reference words:

```
REF:      GREAT OKAY NOW WE COULD GO FROM SAY  -
     - MONTREAL TO WASHINGTON
HYP:  I'M GREAT OKAY NOW WEEK IT  GO FROM CITY -
     - MONTREAL TO WASHINGTON
```

In addition, speech recognizers are increasingly being used as "black-boxes," having a clearly specified function and well-defined inputs and outputs but otherwise providing no hooks for altering or tuning internal operations, with the notable exception of the ability to add words to the recognizer's vocabulary. As an example of speech recognition as a black-box, several research labs have announced plans to make speech recognition available to the research community by running publicly accessible speech servers on the Internet. Such servers would likely employ a general-purpose language model and acoustic model. In order to employ them for a task involving words not available to the server's language model, a remote user would need some way to correct the errors committed by the black-box SR server.

This paper presents a new technique for overcoming several types of speech recognition errors by post-processing the output of a continuous speech recognizer. The post-processor output contains fewer errors, thereby making interpretation by higher-level modules, such as a parser, in a speech understanding system more reliable. The goal of this work is to contribute to successful understanding of spontaneous spoken utterances in human-computer dialogue by a conversational planning assistant called the TRAINS-95 system.

Our objective is to reduce speech recognition errors by refining or even modifying the effective vocabulary of a speech recognizer. To achieve this, we regard the channel from the speaker to the output of the SR module as a noisy channel, and we adopt statistical techniques (some of them borrowed from statistical machine translation) for modeling that channel in order to correct some of the errors introduced there.

Why reduce recognition errors by post-processing the SR output? Why not simply better tune the SR's language model for the task? First, if the SR is a general-purpose black-box (running either locally or on the other side of a network on someone else's machine), modifying the decoding algorithm to incorporate the post-processor's model might not be an option. Using a general-purpose SR engine makes sense because it allows a system to deal with diverse utterances. If needed, the post-processor can tune the general-purpose hypothesis in a domain-specific or user-specific way (there is also room for adapting to domains and users on-line if the engine was not designed to do so). Porting an entire system to new domains only requires tuning the post-processor, and the general-purpose component with its models can be reused with little or no change. Be-

*THIS WORK WAS SUPPORTED BY THE UNIVERSITY OF ROCHESTER CS DEPARTMENT AND ONR/ARPA RESEARCH GRANT NUMBER N00014-92-J-1512.

[1] For this experiment involving Sphinx-II, the acoustic model and the class-based language model were trained on ATIS data. Hence, some of the error is attributable to the moderate occurrence of out-of-vocabulary (OOV) words.

[2] For this experiment involving the HTK-based recognizer, the acoustic model and the word-based language model were trained on the Trains Dialogue Corpus [6] (collected prior to the creation of the TRAINS-95 system).

[3] In the examples, the `HYP` tag indicates the SR system's *hypothesis*, and the `REF` tag indicates the *reference* transcription.

cause the post-processor is light-weight by comparison, the savings may be significant.

Second, even if the SR engine's language model can be updated with new domain-specific data, the post-processor trained on the *same* new data can provide additional improvements in accuracy.

Third, several human speech phenomena are poorly modeled by current continuous SR technology, and recognition is accordingly impaired. This suggests that the SR module does indeed belong as a component of the noisy channel. One poorly modeled phenomenon is *assimilation* of phonetic features. Most SR engines model phonemes in a context-dependent fashion (*e.g.*, see [10]), and some attempt to model cross-word co-articulation effects (*c.f.* [10] also). However, as speaking speeds vary, the SR's models may not be well suited to the affected speech signal. Such errors can be corrected by the post-processing techniques discussed here.

Finally, the primary advantage to the post-processing approach over existing approaches for overcoming SR errors lies in its ability to introduce options that are not available in the SR module's output. Existing rescoring tactics cannot do so (*c.f.* [4, 12]).

2. THE MODELS AND ALGORITHM

A statistical model for automatically translating individual sentences between two human languages was proposed by Brown *et al.* [3]. While this approach to translation has its critics, we can adapt the same idea to the process of transcribing a spoken utterance. We simply posit the existence of a string of English words ($\underline{\mathbf{e}}_{1,n} = (w_1, w_2, \ldots, w_n)$) in the mind of the speaker. Those words are uttered and transmitted to the listening system's microphone. The sounds are then transcribed as a string of English words ($\underline{\mathbf{e}}'_{1,n'}$) by the SR component of the system. The channel beginning at the speaker and ending at the output of the SR module is a noisy channel, in which errors are frequently introduced in all segments of the channel, including the SR module, essentially at the word-level. We adapt the statistical MT techniques to recover the original string of words and thereby correct some of the errors introduced in the channel. Figure 1 illustrates the relationship of the speaker, the channel, and the error-correcting post-processor.

Brown *et al.* delineate their approach into three parts: a translation (or channel) model, a language model, and a search among possible source word sequences. We will describe each component for our approach to SR post-processing.

We adopt a channel model that describes some of the effects on utterances that pass through the noisy channel ending with the speech recognizer. Specifically, it accounts for frequent errors such as simple word/word confusions and short phrasal and segmentation problems (*e.g.*, one-to-many word substitutions and many-to-one word concatenations). In addition to the channel model, we present a suitable search algorithm that uses the model (together with a source language model) to find the most likely correction for a given word sequence from the SR module. We have built a post-processor that employs these models and have wedged it into the interpretation pipeline of the TRAINS-95 system just behind the SR module. This implementation of the post-processor can receive input from the SR module incrementally as the SR decoder improves its primary hypothesis. The post-processor also communicates with the TRAINS-95 parser in an incremental fashion, backing up occasionally where partial solutions change on the fly.

The post-processor repairs utterances according to the probability estimates acquired from training data. If the training set consists of words from a task-specific vocabulary, then the post-processor will map the general-purpose vocabulary of the SR module to task-specific vocabulary. If the training set consists of words from *another* domain, then the post-processor will map the SR vocabulary to the vocabulary of the other domain. If the recognizer suggests a word that was not observed as a misrecognition in the post-processor's training set, then the post-processor will simply forward the unknown word to subsequent components. If, however, that word is known to be frequently misrecognized, then the post-processor will correct it to the appropriate in-domain word.

By applying Bayes' rule, we derive a simple expression for the most likely pre-channel sequence $\hat{\underline{\mathbf{e}}}_{1,n}$. The derivation is similar to the derivation of the statistical approach to SR (as explained in [2, 8]):

$$\hat{\underline{\mathbf{e}}}_{1,n} = \operatorname*{argmax}_{\underline{\mathbf{e}}_{1,n}} \mathrm{P}[\underline{\mathbf{e}}_{1,n}] \cdot \mathrm{P}[\underline{\mathbf{e}}'_{1,n'} \mid \underline{\mathbf{e}}_{1,n}] \ . \qquad (1)$$

The first factor, $\mathrm{P}[\underline{\mathbf{e}}_{1,n}]$, models the formation of English utterances by the speaker. It is the listener's model of the speaker's language. The second factor, $\mathrm{P}[\underline{\mathbf{e}}'_{1,n'} \mid \underline{\mathbf{e}}_{1,n}]$, models the behavior of the channel.

2.1. First Approximation

For a sizable vocabulary, adequately estimating the probability distributions that model the channel and the speaker's language requires mammoth amounts of data; therefore, it is necessary to approximate through independence assumptions. Several assumptions are possible, and we will begin with a basic set of assumptions before suggesting others.

For a first approximation language model, we use a word-bigram model.

$$\hat{\mathrm{P}}[\underline{\mathbf{e}}_{1,n}] = \prod_{i=0}^{n-1} \mathrm{P}[w_{i+1} \mid w_i] \ . \qquad (2)$$

As a first approximation channel model, we assume that each word in $\underline{\mathbf{e}}'_{1,n'}$ is simply a transmitted version of the word with the corresponding position in $\underline{\mathbf{e}}_{1,n}$. Thus,

$$\hat{\mathrm{P}}[\underline{\mathbf{e}}'_{1,n'} \mid \underline{\mathbf{e}}_{1,n}] = \prod_{i=1}^{n} \mathrm{P}[w'_i \mid w_i] \ . \qquad (3)$$

We say that a word is *aligned* with the word it produces.

We also require a method for searching among possible source utterances $\underline{\mathbf{e}}_{1,n}$ for the most likely correction of the given word sequence, *i.e.*, the one that yields the greatest value of $\mathrm{P}[\underline{\mathbf{e}}_{1,n}] \cdot \mathrm{P}[\underline{\mathbf{e}}'_{1,n'} \mid \underline{\mathbf{e}}_{1,n}]$. We use a Viterbi beam-search for this purpose (*c.f.* [5, 11]).

2.2. Enhancements to the Models

To improve the language model, we use higher-order n-grams, thereby assuming that each word in $\underline{\mathbf{e}}_{1,n}$ is dependent on its $n-1$ predecessors. We also use back-off n-gram models for combating the problem of sparse training data [9].

For the channel model, we relax the constraint that replacement errors be aligned on a word by word basis, since not all recognition errors consist of simple replacement of

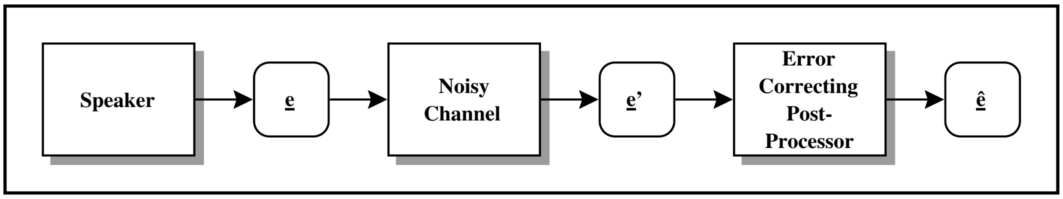

Figure 1. *Recovering Word-Sequences Corrupted in a Noisy Channel.*

one word by another. Some errors appear as the break-up of one word into shorter words. Other errors involve the erroneous concatenation of two or more words to make a longer word. We will use the following utterance from the TRAINS-95 dialogues as an example.

```
REF:  GO FROM CHICAGO TO TOLEDO
HYP:  GO FROM CHICAGO TO TO LEAVE AT
```

Following Brown *et al.*, we refer to a picture such as Figure 2 as an *alignment*. We use an *alignment* to indi-

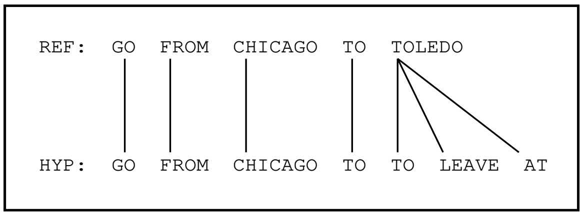

Figure 2. *Alignment of a Hypothesis and the Reference Transcription.*

cate the source words in the REF sequence for each of the words in the HYP sequence. For alignments, we use the following notation: we write the post-channel transcription ($\underline{\mathbf{e}}'_{1,n'}$) followed by the pre-channel transcription ($\underline{\mathbf{e}}_{1,n}$) separated by a vertical bar and enclosed in parentheses. We also refer to the number of post-channel words produced by a pre-channel word in a particular alignment as the *fertility* of that pre-channel word. Following each of the pre-channel words, we provide its fertility in the current alignment in parentheses. Alignments are easily computed using a dynamic programming algorithm for word sequence alignment. Returning to our example, we have the alignment:

```
(GO FROM CHICAGO TO TO LEAVE AT
  | GO(1) FROM(1) CHICAGO(1) TO(1) TOLEDO(3))
```

To augment our channel model, we require a *fertility model* $\mathrm{P}[k \mid w]$ that indicates how likely each word w in the pre-channel vocabulary will have a particular fertility k. When a word's fertility k is an integer value between two and five, [4] it indicates that the pre-channel word resulted in multiple post-channel words. When a word's fertility is one, then the word accounts for exactly one post-channel word. When a word's fertility is a fraction $\frac{1}{n}$ (for $2 \leq n \leq 5$), then the word and $n-1$ neighboring words have grouped together to result in a single post-channel word. We call this situation *fractional fertility*. For example, a word with $k = \frac{1}{3}$ indicates the situation in which this word and two neighboring source words contribute to one word in the hypothesis; *i.e.*, each word accounts for one-third of the post-channel word. When a word's fertility is a fraction $\frac{m}{n}$ (for $2 \leq m \neq n \leq 5$), then the word and $n-1$ neighboring pre-channel words have grouped together to result in m post-channel words. The latter case can be used to handle arbitrary segmentation errors. For example, a word with $k = \frac{3}{2}$ indicates that this word and a neighboring source word contribute to three words in the hypothesis; thus, we can imagine each word accounting for three-halves of the post-channel words. A concrete example of this alignment is `(TO LEAVE DOING | TOLEDO(3/2) IN(1/2))`.

[4] Values higher than five are ignored, since they are very rare.

To understand how fertility models are used, we need to extend the basic search algorithm. As before, the algorithm searches for an optimal source utterance $\underline{\mathbf{e}}_{1,n}$, modulo the beam pruning. This extended search builds possible sequences $\underline{\mathbf{e}}_{1,n}$ one word at a time using $\underline{\mathbf{e}}'_{1,n'}$ for guidance as before. Each word in $\underline{\mathbf{e}}'_{1,n'}$ is exploded (or collapsed with neighbors) using all possible combinations. The hypotheses are scored according to 1. the LM and 2. the channel model for one-for-one replacements or the fertility model for other kinds of replacements. As before, dynamic programming on partial source sentences and beam pruning will make the search efficient.

Observe that the fertility model scores only *the number of words* used to replace a particular word. It actually relies on the language model to score the contents of the replacement. This is motivated by the related approach of Brown *et al.*, who appear to have taken this direction in order to avoid the problems of gathering statistics from hopelessly sparse data.

3. EXPERIMENTAL RESULTS

The post-processor has been implemented to use the simple one-for-one channel model and a back-off bigram language model. The channel model incorporating fertility is work in progress. The language model was trained on hand-transcribed utterances from the TRAINS-95 dialogues. The channel model was constructed by automatically aligning the output of Sphinx-II (having fixed language and acoustic models) with the hand transcriptions and by tabulating substitutions.

To test the post-processor, an independent set of utterances was held out for evaluation. The cross-validated performance of Sphinx-II alone and in tandem with the Post-processor are depicted in Figure 3. Sphinx-II's class-based language model was trained only on data from the ATIS spoken language corpora. Also illustrated are the amounts of training data required by the post-processor to make a particular contribution to word recognition accuracy. This validates the claim that the post-processor can make a significant impact in tuning the SR if the SR cannot be modified as we have discussed. Also, equivalent amounts of training data can be used with comparable impact in the post-processor as in the language model of the SR. Furthermore, preliminary results indicate that if the language model of the SR can indeed be modified, then the post-processor can still significantly improve word recognition accuracy. Hence the post-processor is in neither case redundant.

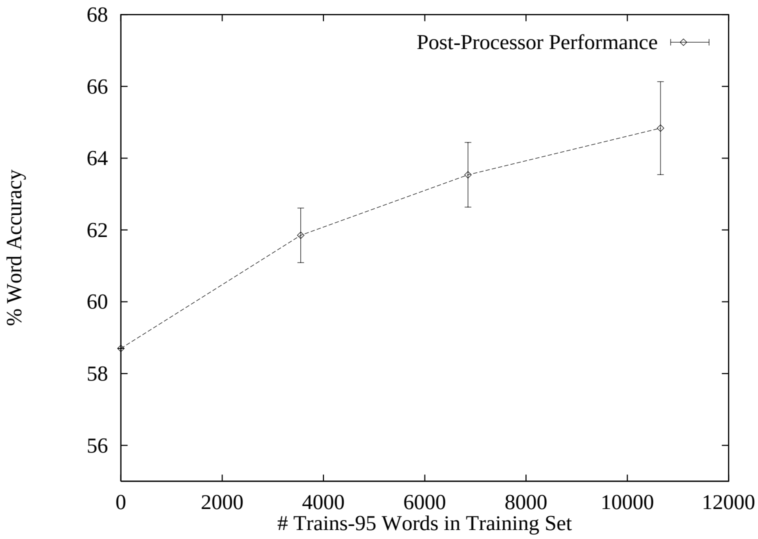

Figure 3. *Influence of the post-processor with additional training data.*

4. FUTURE DIRECTIONS

We have presented models and methods for overcoming speech recognition errors. We have also provided evidence for the claim that modern speech recognition engines can be used successfully as black-boxes for robustly interpreting utterances in a dialogue with a human.

Open issues include whether word-lattices will provide better opportunities over simple word sequences for post-processor correction. For the word-lattice configuration, the post-processor must be modified to process the alternatives in the lattice. One point to consider here is the width of the lattice (*i.e.*, the number of alternatives at a given point in the utterance). This factor can implicitly reflect the confidence of the SR in its hypotheses and may be useful as a parameter in the correction process.

In addition to the purely statistical mechanisms for recovering pre-channel word sequences outlined above, other cues may augment the search. For example, syllables and vowel nuclei may be usable for aligning pre-channel and post-channel words and phrases. Such alignments may be useful for further constraining the search algorithms and yielding better corrections.

REFERENCES

[1] J. F. Allen, G. Ferguson, B. Miller, and E. Ringger. Spoken Dialogue and Interactive Planning. In *Proceedings of the ARPA SLST Workshop*, San Mateo California, January 1995. ARPA, Morgan Kaufmann.

[2] L. R. Bahl, F. Jelinek, and R. Mercer. A Maximum Likelihood Approach to Continuous Speech Recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, 5(2):179–190, March 1983.

[3] P. F. Brown, J. Cocke, S. A. Della Pietra, V. J. Della Pietra, F. Jelinek, J. D. Lafferty, R. L. Mercer, and P. S. Roossin. A Statistical Approach to Machine Translation. *Computational Linguistics*, 16(2):79–85, June 1990.

[4] Y. Chow and R. Schwartz. The n-best algorithm: An efficient procedure for finding top n sentence hypotheses. In *Proceedings of the Second DARPA Workshop on Speech and Natural Language*, pages 199–202, San Mateo, California, October 1989. DARPA, Morgan Kaufmann.

[5] J. G. E. Forney. The Viterbi Algorithm. In *Proceedings of IEEE*, volume 61, pages 266–278. IEEE, 1973.

[6] P. Heeman and J. F. Allen. The TRAINS 93 Dialogues. TRAINS Technical Note 94-2, Department of Computer Science, University of Rochester, Rochester, NY, 14627, March 1995.

[7] X. D. Huang, F. Alleva, H. W. Hon, M. Y. Hwang, K. F. Lee, and R. Rosenfeld. The Sphinx-II Speech Recognition System: An Overview. *Computer, Speech and Language*, 1993.

[8] F. Jelinek. Self-Organized Language Modeling for Speech Recognition. Reprinted in [13]: 450-506, 1990.

[9] S. M. Katz. Estimation of probabilities from sparse data for the language model component of a speech recognizer. In *IEEE Transactions on Acoustics, Speech, and Signal Processing*, pages 400–401. IEEE, IEEE, March 1987.

[10] K.-F. Lee. *Automatic Speech Recognition: the Development of the SPHINX System.* Kluwer Academic, Boston, London, 1989.

[11] B. Lowerre and R. Reddy. The Harpy Speech Understanding System. In *Trends in Speech Recognition.* Speech Science Publications, Apple Valley, Minnesota, 1986. Reprinted in [13]: 576-586.

[12] M. Rayner, D. Carter, V. Digalakis, and P. Price. Combining Knowledge Sources to Reorder N-best Speech Hypothesis Lists. In *Proceedings ARPA Human Language Technology Workshop*, pages 212–217. ARPA, March 1994.

[13] A. Waibel and K.-F. Lee, editors. *Readings in Speech Recognition.* Morgan Kaufmann, San Mateo, 1990.

[14] S. J. Young and P. C. Woodland. *HTK: Hidden Markov Model Toolkit.* Entropic Research Laboratory, Washington, D.C., 1993.