ON-LINE LEARNING OF LANGUAGE MODELS WITH WORD ERROR PROBABILITY DISTRIBUTIONS

Roberto Gretter[*]

ITC-irst, Povo di Trento, ITALY
gretter@itc.it

Giuseppe Riccardi

AT&T Labs-Research, Florham Park, NJ, USA
dsp3@research.att.com

ABSTRACT

We are interested in the problem of learning stochastic language models on-line (*without* speech transcriptions) for adaptive speech recognition and understanding. In this paper we propose an algorithm to adapt to variations in the language model distributions based on the speech input only and without its true transcription. The on-line probability estimate is defined as a function of the prior and word error distributions. We show the effectiveness of word-lattice based error probability distributions in terms of Receiver Operating Characteristics (ROC) curves and word accuracy. We apply the new estimates $P_{adapt}(w)$ to the task of adapting on-line an initial large vocabulary trigram language model and show improvement in word accuracy with respect to the baseline speech recognizer.

1. INTRODUCTION

We are interested in the problem of learning stochastic language models on-line (*without* speech transcriptions) for adaptive speech recognition and understanding. Currently, speech understanding systems are designed for a given application and trained on the domain-specific speech and text corpus. On-line learning is critical to enable spoken dialog systems to shift the goal of existing tasks or adapt to new domains.

In this paper we propose an algorithm to adapt to variations in the language model distributions based on the speech input only and without its transcription. Since the speech recognizer output is noisy we need an error probability distribution to account for the input noisy channel and complete the probabilistic model:

$$P_{adapt}(w) = F(P_o(w), P_{error}(w)) \qquad (1)$$

where $P_{adapt}(w)$ is the on-line adapted word distribution, F is a function of $P_o(w)$, the initial probability distribution of word w and of $P_{error}(w)$, the error probability distribution.

[*]The author performed the work while at AT&T Labs-Research, Florham Park, NJ.

In the literature there are two types of word error probability distributions. The first type is based on acoustic measurements [1] and the other type on word lattices. The latter has the advantage that the probabilities are computed on-line and does not require training. In particular, *sausages* [2] are compact representations for word lattices. *Sausages* topology is such that posterior probabilities are easily estimated and are effective predictors for word accuracy. In the next paragraph we review the *sausage* lattice representation and provide experimental data for the word error distribution and word accuracy results on a large vocabulary task. In the second paragraph, we describe the on-line experiments performed with the goal of learning $F(P_o(w), P_{error}(w))$. We apply the new estimates $P_{adapt}(w)$ to the task of adapting on-line an initial trigram language model from untranscribed speech data and show improvement in word accuracy with respect to the baseline speech recognizer.

2. LATTICES AND SAUSAGES

One of the possible outputs of a speech recognizer is a word lattice. A word lattice is a connected graph, where each state has a time information and each arc represents a word that has been hypothesized during the decoding. Each arc has a score, which comes from the combination of the acoustic and language models; the topology of the lattice reflects the constraints of the language model.

Recently, an algorithm has been proposed [2] for converting a lattice in a compact format, called *sausage*. A sausage is a simplified lattice with a particular topology: it turns out to be a sequence of confusion sets, each one being a group of words, which may include a null word (*eps*), competing in the same (with some approximation) time interval. Each word has a posterior probability, which is the sum of the probabilities of all the paths of that word occurrence in the lattice. In each confusion set, the sum of all posteriors equals 1. In a sausage the time order is preserved, but time information is lost. The main motivation of this algorithm is that of minimizing the Word Error Rate (WER), instead of the Sentence Error Rate (SER).

We briefly review Mangu's algorithm. Mangu's algo-

rithm takes a word lattice as input and goes through the following steps:

- low probability links of the lattice are pruned;
- a posterior probability for each link of the lattice is computed;
- a "temporal" order over the states of the lattice is found;
- different occurrences of the same word in the "same" time interval are merged (intra - word clustering stage) and their posteriors summed;
- words which compete in the "same" time interval are grouped together to form a confusion set (inter - word clustering stage).

It is straightforward to extract from a sausage what is called the *Consensus Hypothesis*, which is the word sequence obtained by picking up from each confusion set the word with the highest posterior. This sequence can differ from the best path hypothesis, i.e. the optimal word sequence inside the lattice. The Consensus Hypothesis is said to have a better WER of the best path hypothesis, and experiments on the HMIHY task ([3]) confirm this. In fact, on the 1K test set, a gain of 0.6% was obtained[1], as shown in Table 1.

	CORR	WA	INS	DEL	SUB
best path	67.8%	62.6%	5.2%	12.0%	20.2%
consensus	68.1%	63.2%	4.9%	12.5%	19.4%

Table 1. Consensus Hypothesis gain over Best Path, for the HMIHY task.

3. LATTICE-BASED POSTERIOR PROBABILITY

As previously said, each word of the sausage (and in particular the words of the Consensus Hypothesis) has associated a posterior probability. In Table 2 an example is reported for the sausage of Figure 1.

ref	yeah	I'm	making	a		credit	card	call
b_p	yeah	I'm	making	a	yet	credit	card	call
cns	yeah	I'm	making	a	yet	credit	card	call
pst	.995	.994	.997	.997	.622	.994	.998	.999

Table 2. Reference, Best Path, Consensus Hypothesis and Posteriors for an HMIHY sentence. In this case Best Path and Consensus Hypothesis are the same, but the posterior is a good indicator of a possible speech recognition error.

[1] In this experiment, lattices did not contain time information.

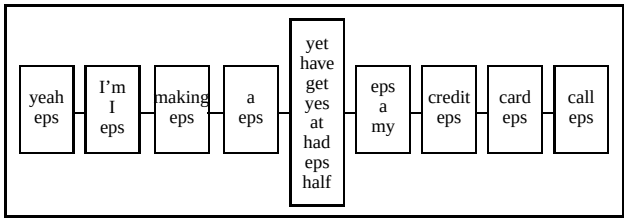

Fig. 1. A sausage. Note the empty word *eps*.

Another quantity that can be used as a confidence score is a local entropy, computed on each confusion set:

$H = -\sum_{i=1}^{N} post(w_i) \times log(post(w_i))$

where N is the number of competing hypotheses and $post(w_i)$ is the posterior of word w_i. This measure includes more information than the previous one, because it takes into account not only the posterior of the winning word, but also the distribution of the posteriors of the competing words.

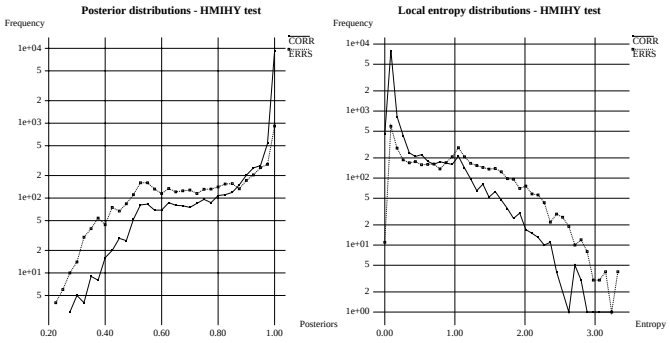

Fig. 2. Distributions for posteriors and local entropy, for words correctly recognized (solid lines) and for words misrecognized (dotted lines).

Posteriors and local entropy have been computed on the HMIHY test set. Figure 2 shows their distributions for words that have been correctly recognized and for words that have been misrecognized. In both cases there is separability among distributions. Thus the posterior appears to be a good candidate as a confidence score. The posterior is then interpreted as the probability of correctly recognizing a word w $(1 - P_{error}(w))$.

Then a ROC curve was drawn for both quantities, showing a very similar behavior. Figure 3 shows false rejection rate

$(100 \times (1 - correct/occurrences))$

against correct classification rate

$(100 \times (correct/hypothesized))$

for the HMIHY test set.

Both posterior and local entropy give a measure of the acoustic / linguistic confusion found during the decoding

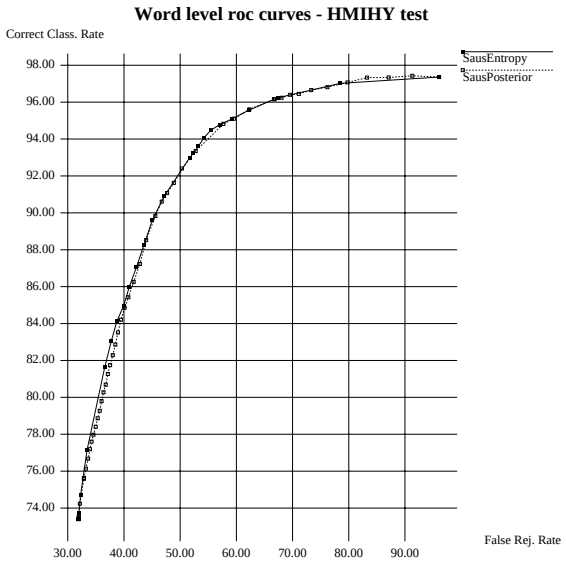

Fig. 3. ROC curves.

stage; in this sense, they can identify speech segments with problematic speech recognition. Compared to other measures, like the likelihood ratios ([1]), they are robust toward changes in the speech recognizer and do not need training. As far as implementation issues are concerned, they can be computed as a simple post processing of the recognizer's output, without need for a second recognition stage.

4. UNSUPERVISED ON-LINE LEARNING

A set of experiments were performed to verify the possibility of using the posteriors of the Consensus Hypothesis to improve the performance of a system in an unsupervised way.

For this experiment we used the following databases from the *How May How Help You?* task [3]:

- Tr (HMIHY '95, 7844 sentences): speech and transcription corpus used to train the acoustic and language models; this corpus is a collection of speech transcriptions from human-human interactions.
- Tr^1 (HMIHY '97, 6896 sentences): speech corpus used to adapt the language models; this corpus is a collection of speech transcriptions from human-machine interactions. The corpora Tr and Tr^1 have significantly different language distributions [3].
- $Test$: (HMIHY '97, 899 sentences): test corpus from the same collection as the set Tr^1.

The goal here is that of using only the speech data of Tr^1, without transcriptions, to modify the language models of the baseline, trying to improve the word accuracy on

true transcription sequence
to Paul Barlett
collect to Kayme
-irectly to the number I'm calling from thank you
after <UNKNOWN> replacing
to <UNKNOWN> <UNKNOWN>
collect to <UNKNOWN>
<UNKNOWN> to the number I'm calling from thank you

Table 3. Upperbound: the label <UNKNOWN> replaces the words outside the initial lexicon

$Test$. The set of sentences on which the language model (trigrams) will be trained is the sole part of the system that will change during the experiments: lexicon, acoustic models and other parameters (language model weight, insertion penalties, etc.) will be kept fixed.

The baseline, an upperbound, and a first experiment (best path) for this data have been obtained by training the language model on the following data:

- baseline: the training set is only Tr;
- best path: the training set is the concatenation of Tr and the output of the recognizer (baseline) on Tr^1;
- upperbound: the training set is composed by Tr and Tr^1: this is the unique experiment in which the manual transcriptions of Tr^1 have been used. Note that, in order to keep the lexicon fixed, words not in the original lexicon (that of Tr) were replaced by the label <UNKNOWN> in the training data, as shown in Table 3. Then, after trigram estimation, all the paths using the label <UNKNOWN> were removed from the grammar.

	perpl	WA	INS	DEL	SUB
baseline	17.60	58.5%	12.6%	5.7%	23.3%
best path	14.73	58.6%	13.2%	5.1%	23.1%
upperbound	14.01	60.7%	12.1%	5.4%	21.8%

Table 4. Results for the baseline

Table 4 reports results obtained in these experiments, expressed both in terms of perplexity on the test set and in Word Accuracy (WA). Note that, while in both cases (best path and upperbound) there is a significant decrease in perplexity with respect to the baseline, recognition performance improves only for the upperbound: adding the output of the recognizer to the initial training set does not give benefits with respect to the baseline. In the next experiment

consensus hypothesis with posteriors
dial(0.46) the(0.29) call(0.96) calling(0.95) card(0.94) call(0.91) my(0.84) phone(0.76) bill(0.58) yes(0.66) calling(1) card(1) six(0.99) six(0.52) oh(0.71)
corresponding training set
<UNUSED> <UNUSED> call calling card call my phone <UNUSED> <UNUSED> calling card six <UNUSED> oh

Table 5. Posterior thresholding: the label <UNUSED> replaces the words with a low posterior (<0.7)

we used the posteriors for the Consensus Hypothesis, marking as <UNUSED> all the words of Tr^1 having a posterior below a given threshold. This is shown in the sample of Table 5. By varying the threshold, the percentage of words of Tr^1 marked as <UNUSED> changes, modifying therefore the training set. The probability estimate $P_{adapt}(w)$ is computed from the count $C_{adapt}(w)$:

$$C_{adapt}(w) = C_{Tr}(w) + C_{Tr^1}(w, tsh) \qquad (2)$$

where $C_{Tr}(w)$ is the number of occurrence of word w in the corpus Tr, and $C_{Tr^1}(w, tsh)$ is the number of occurrences of word w with error probability less than tsh. The count $C_{adapt}(w)$ is used in the back-off probability estimate formula for n-grams in a straightforward manner. This allows to obtain different perplexities and performances, as shown in Table 6. In particular, while the perplexity monotonically increases with the percentage of unused words, the word accuracy has a maximum when about half of the words of Tr^1 are used to train the language model.

thr	unused	perpl	WA	INS	DEL	SUB
0.50	7.38%	14.69	58.8%	13.0%	5.5%	22.7%
0.70	20.25%	14.71	58.9%	12.7%	5.7%	22.6%
0.90	35.68%	14.76	58.7%	12.6%	5.7%	23.0%
0.95	42.39%	14.85	59.1%	12.3%	5.8%	22.8%
0.98	50.00%	15.13	59.1%	12.2%	5.8%	22.9%
0.99	55.16%	15.33	59.0%	12.3%	5.9%	22.8%
.999	73.33%	15.73	58.7%	12.3%	6.0%	23.0%
1.0	82.90%	15.91	58.8%	12.2%	5.8%	23.1%

Table 6. Posterior thresholding: results for different values of the threshold.

The last experiment was performed by using the posteriors directly in the trigram computation, i.e. by incrementing the counters for the trigram estimation by the posteriors instead of 1. The estimate for the word counts is then:

$$\begin{aligned} C_{adapt}(w) &= C_{Tr}(w) + \sum_{w \in Tr^1} (1 - P_{error}(w)) \qquad (3) \\ &= C_{Tr}(w) + C_{Tr^1}(w) - \sum_{w \in Tr^1} P_{error}(w) \end{aligned}$$

In this way all the words of the consensus hypotheses were used during training, but weighted by their confidence scores. To be more precise, the training set in this case was composed by the transcriptions of Tr and the consensus hypothesis computed on Tr^1; the words of Tr (for which we used the true transcriptions) incremented the counters by 1, while the words of Tr^1 incremented the counters by their posterior. This leads (see Table 7) to a low perplexity (14.67) and good WA (59.0%).

	perpl	WA	INS	DEL	SUB
counters	14.67	59.0%	12.9%	5.5%	22.6%

Table 7. Results using posteriors as counters in trigram estimation.

5. CONCLUSIONS

On-line learning of language models is crucial for creating adaptive spoken dialog systems. In this paper we have proposed an algorithm to adapt to variations in the language model distributions based on the speech input. The on-line probability estimate is defined as a function of the prior and word error distributions. We have shown the effectiveness of word-lattice based error probability distributions in terms of Receiver Operating Characteristics (ROC) curves and word accuracy. We have investigated two update schemes for the on-line probability estimates. We have applied the new estimates to the task of adapting on-line an initial large vocabulary trigram language model and shown improvement in word accuracy with respect to the baseline speech recognizer.

Acknowledgment We would like to thank Lidia Mangu for providing the sausage computation software.

6. REFERENCES

[1] R. C. Rose, B. H. Juang, and C. H. Lee, "A Training Procedure for Verifying String Hypotheses in Continuous Speech Recognition," in *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing*, Detroit, USA, 1995, pp. 281–284.

[2] L. Mangu, E. Brill, and A. Stolcke, "Finding Consensus Among Words: Lattice-Based Word Error Minimization," in *Proceedings of the European Conference on Speech Communication and Technology*, Budapest, Hungary, 1999.

[3] G. Riccardi and A. L. Gorin, "Stochastic Language Adaptation Over Time and State in a Natural Spoken Dialog System," in *IEEE Trans. on Speech and Audio Proc*, vol.8, no. 1, January 2000.